Calhoun: The NPS Institutional Archive

DSpace Repository

Theses and Dissertations 1. Thesis and Dissertation Collection, all items

1972

Application of discrete Walsh functions to digital filter techniques.

Meck, Norman Clare.

Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/15992

Downloaded from NPS Archive: Calhoun

Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

http://www.nps.edu/library

APPLICATION OF DISCRETE WALSH FUNCTIONS TO DIGITAL FILTER TECHNIQUES

Norman Clare Meck

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

Application of Discrete Walsh Functions to Digital Filter Techniques

by

Norman Clare Meck

Thesis Advisor: S. R. Parker

June 1972

T147796

Approved for public release; distribution unlimited.

Application of Discrete Walsh Functions to Digital Filter Techniques

by

Norman Clare Meck
Lieutenant, United States Navy
B.S., University of Colorado, 1965

Submitted in partial fulfillment of the
requirements for the degree of

ELECTRICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL
June 1972

ABSTRACT

Methods of generating and properties of Walsh functions are discussed with emphasis on discrete Walsh functions. Hardware and software versions of two methods of Walsh function generation are shown. The Walsh transform is explored and hardware and software realizations of a fast Walsh transform algorithm, particularly applicable to hardware, is introduced. Computer programs containing the various algorithms are included. A discussion of the feasibility of the use of Walsh functions to produce a digital filter with a desired frequency response is presented.

TABLE OF CONTENTS

I. HISTORICAL BACKGROUND AND SCOPE OF THESIS -- 6

II. PROPERTIES OF WALSH FUNCTIONS -----------------10

III. GENERATION OF WALSH FUNCTIONS -----------------15

IV. THE WALSH TRANSFORM ---------------------------18

V. DIGITAL FILTER CONSIDERATIONS -----------------24

VI. RESULTS AND CONCLUSIONS -----------------------31

APPENDIX A - WALSH FUNCTIONS AS PRODUCTS OF RADEMACHER FUNCTIONS -----------------33

APPENDIX B - FREQUENCY SPECTRUM (FOURIER TRANSFORM) OF THE CONTINUOUS WALSH FUNCTIONS ------36

COMPUTER PROGRAM 1 - HADAMARD MATRIX GENERATION -----55

COMPUTER PROGRAM 2 - WALSH TRANSFORM AS HADAMARD MATRIX PRODUCT ------------------60

COMPUTER PROGRAM 3 - FAST WALSH TRANSFORM -----------68

BIBLIOGRAPHY --76

INITIAL DISTRIBUTION LIST ---------------------------78

FORM DD1473 ____79

LIST OF ILLUSTRATIONS

1. The First 16 Harr Functions, $\chi_k^{(i)}(t)$ ------------- 38

2. The First 5 Rademacher Functions, R(k,t) --------- 39

3. The First 16 Walsh Functions, W(k,t) ------------- 40

4. Discrete Walsh Functions of Order 16 (Sequency Ordered) ----------------------------- 41

5. Walsh Function Generator (as Rademacher Function Products) ----------------------------- 42

6. Generation of $W_{16}(11,t)$ ------------------------- 43

7. Serial In/Parallel Out Fast Walsh Transform Realization ---------------------------------- 44

8. Fast Walsh Transform Realization Parallel In/Parallel Out ------------------- 45

9. Discrete Walsh Functions of Order 16 (Binary Ordered) ------------------------------ 46

10. Frequency Spectrum of the First 8 Discrete Walsh Functions, $W_{16}(k,j)$ ------------------- 47

11. Frequency Spectrum of the Second 8 Discrete Walsh Functions, $W_{16}(k,j)$ ------------------- 48

12. Walsh Function Resonator ------------------------ 49

13. Walsh Function Resonator for $W_{32}(14,j)$ ----------- 50

14. Cascade Version of Walsh Resonator --------------- 51

15. Cascade Version of $W_{32}(14,j)$ Walsh Resonator ----- 52

16. Narrow Bandpass Filter Using Walsh Resonators ---- 53

17. Frequency Response of Narrow Bandpass Filter Using Walsh Resonators ---------------------- 54

ACKNOWLEDGMENT

The assistance and advice of William J. Dejka, Modular Concepts Division Head, Naval Electronics Laboratory Command, is gratefully acknowledged.

I. HISTORICAL BACKGROUND AND SCOPE OF THESIS

A pair of functions are said to be orthogonal if the integral of the product of those functions over all space vanishes. Systems of orthogonal functions have been available for many years. At the beginning of the twentieth century, there were many well known and useful orthogonal functions: the trigonometric functions which occur in Fourier series; orthogonal polynomials such as those of Legendre, Hermite, and LaGuerre; Bessel's functions; the Sturm-Lioville series; and other special functions. The general theory including all such systems of functions was essentially developed prior to 1910 and it was shown that a set of such functions is by no means exceptional but can be produced at will.

Alfred Harr [Ref. 10] formulated an orthogonal system of functions in 1909 (now called the Harr functions) which essentially take on only two values. An exemplary set of Harr functions is shown in Figure 1. A formal expansion of an arbitrary continuous function in terms of Harr functions converges uniformly to the given function. This combination of convergence and bi-valued properties had not been possessed by any other orthogonal set known to that time.

Independently, and at about the same time in early 1922, H. Rademacher [Ref. 20] and J.L. Walsh [Ref. 24], each exhibited a set of orthogonal functions taking on the values ± 1 on the interval [0,1]. The Rademacher set, [Figure 2], was

not complete as other non-trivial functions exist which are orthogonal to all of the members of the set. The Walsh functions, [Figure 3], are a complete set and exhibit similarities to the trigonometric functions in many of their properties. The Walsh functions are discussed in detail in the remainder of this thesis.

Very little work was done with these various sets of bi-valued functions until the digital computer became a useful tool of the scientific community. Because of the dual value characteristics of the Harr, Rademacher, and Walsh functions, the applicability to the binary configured machines seemed natural. A substantial amount of effort has been directed into application areas during the past several years and progress has been made in many fields which in the past have only been associated with trigonometric functions. Some of the Walsh function application areas of interest include: multiplexing, pattern recognition, image processing, electromagnetic radiation, and digital filtering. This paper will primarily discuss the area of digital filtering although many of the results are applicable to other areas of interest.

Previously, analysis of digital filters and implementation has been restricted to Laplace and Fourier methods and hence to trigonometric relationships. The implementation of a digital filter based on Walsh functions offers definite advantages unique to a two valued function but also has distinct disadvantages inherent in the use of a set of discontinuous functions to describe a continuous function. Efforts

expended thus far in the area of Walsh function digital filtering appear to be primarily concerned with obtaining the expansion of a given function in terms of the Walsh functions and the recombination of the weighted Walsh functions to obtain the filtered and reconstituted function. In this thesis the manipulation of the Walsh function representation of the given function will be treated and compared to standard Fourier methods. It will be shown that in general there is no advantage to using Walsh functions for digital filtering when a specific frequency response is desired, but that for certain frequency responses, Walsh functions do give a decided advantage over the Fourier methods.

Section II presents some of the properties of Walsh functions including a modified notation method which, it is believed, is clearer and easier to use in mathematical relationships.

Sections III and IV show methods of Walsh function generation and Walsh transform techniques respectively. Algorithms are presented which are adaptable to either hardware or software and will either generate a set of Walsh functions or will take the Walsh transform of a given function. Serial or parallel input of a sampled function is provided for.

Section V considers the use of Walsh functions to produce a digital filter with a desired frequency response. The frequency spectrum of discrete Walsh functions is derived, and it is shown that in general it is as complex to combine these

to achieve a desired frequency spectrum output as is the case in the Fourier domain.

II. PROPERTIES OF WALSH FUNCTIONS

Walsh functions can be defined over the interval [0,1], in the following manner:

$$W(0,t) = \begin{cases} 1, & 0 \leq t \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

$$W(2k+p,t) = W(k,2t)+(-1)^{k+p}W(k,2t-1) \qquad \text{(II-1)}$$

$$p = 0,1$$

$$k = 0,1,2\ldots$$

When Walsh functions are used for digital filtering, the input function is normally sampled at some rate as are the Walsh functions. In order to discriminate between the discrete (sampled) and the continuous functions and for ease in mathematical manipulation, the following notation is used:

Continuous function notation:

$$W(k,t)$$

$k = 0,1,2,\ldots$	index of Walsh function
$0 \leq t \leq 1$	independent variable

Discrete function notation:

$$W_N(k,j)$$

$k = 0,1,2,\ldots, N-1$	index of Wash function
$N =$ a power of 2	order of set of Walsh functions
$j = 0,1,2,\ldots, N-1$	independent variable

The discrete Walsh functions are formed from the continuous functions by sampling $W(k,t)$ at $t = (j/N)^+$, (i.e., from the right). Figure 4 shows the set of order 16 discrete Walsh functions arranged as a 16 x 16 matrix.

Several other forms of notation have been used for Walsh functions. Some of the most common of these are tabulated in Table 1 for comparison and cross reference purposes.

Discrete $j=0,1..,N-1$	Continuous $0\le t\le 1$	Walsh $0\le t\le 1$ [Ref.24]	Corrington $0\le t\le 1$ [Ref. 7]	Harmuth $-\frac{1}{2}\le\theta\le\frac{1}{2}$ [Ref. 9]
$W_8(0,j)$	$W(0,t)$	$\phi_0(t)$	$\psi_{000}(t)$	$Cal(0,\theta)$
$W_8(1,j)$	$W(1,t)$	$\phi_1(t)$	$\psi_{001}(t)$	$Sal(1,\theta)$
$W_8(2,j)$	$W(2,t)$	$\phi_2^{(1)}(t)$	$\psi_{011}(t)$	$Cal(1,\theta)$
$W_8(3,j)$	$W(3,t)$	$\phi_2^{(2)}(t)$	$\psi_{010}(t)$	$Sal(2,\theta)$
$W_8(4,j)$	$W(4,t)$	$\phi_3^{(1)}(t)$	$\psi_{110}(t)$	$Cal(2,\theta)$
$W_8(5,j)$	$W(5,t)$	$\phi_3^{(2)}(t)$	$\psi_{111}(t)$	$Sal(3,\theta)$
$W_8(6,j)$	$W(6,t)$	$\phi_3^{(3)}(t)$	$\psi_{101}(t)$	$Cal(3,\theta)$
$W_8(7,j)$	$W(7,t)$	$\phi_3^{(4)}(t)$	$\psi_{100}(t)$	$Sal(4,\theta)$

Table 1. Notations for Walsh Functions

These functions form a complete, orthonormal set over the interval [0,1]. The implication of this property is that any absolutely integrable function defined over the interval can be expanded in a series of Walsh functions analogous to the Fourier expansion of such a function.

That is:

$$f(t) = \sum_{k=0}^{N-1} a_k W(k,t) \qquad \text{(II-2)}$$

where:

$$a_k = \int_0^1 f(t)\, W(k,t)\, dt \qquad \text{(II-3)}$$

Assuming $f(j\tau)$ is a sampled representation of $f(t)$, where the sample period, $\tau = 1/N$; then $a_k = 0$ for

$k \geqslant N$ and equations (II-2) and (II-3) reduce to:

$$f(j\tau) = \sum_{j=0}^{N-1} a_k \; W_N(k,j) \qquad \text{(II-4)}$$

and

$$a_k = 1/N \sum_{j=0}^{N-1} f(j\tau) \; W_n(k,j) \qquad \text{(II-5)}$$

If the function $f(t)$ or $f(j\tau)$ is defined over the interval $[0,T]$, a simple change of scale, $t_1 = t/T$ will allow the use of the above equations to find the Walsh representation of the function over $[0,1]$.

Sequency is defined as one-half the number of zero crossings per unit time. The number of zero crossings and hence the sequency of a given Walsh function can be obtained by:

$$z = \frac{k+(-1)^{k+1}}{2} \qquad \text{(II-6)}$$

where z is the sequency

and k is the index of the Walsh function

The set, $\{a_k\}$, therefore represents the sequency spectrum of $f(t)$ in the same sense that a set of Fourier coefficients represents a frequency spectrum. It should be noted that the Walsh functions defined in this thesis are ordered according to increasing sequency. This is the definition as used by Walsh in his original paper [Ref. 24] and is equivalent to that used by Harmuth [Ref. 9]. Another method of defining Walsh functions is as products of Rademacher functions. When defined in this way, they are not ordered according to increasing sequency but are what is termed binary ordered. Appendix A shows this method of defining

Walsh functions and derives algorithms for converting from one method of ordering to the other. The original sequency ordering of Walsh seems to be best suited to the purposes of communications engineering.

Another property of Walsh functions can be seen from their internal symmetries. A product of two Walsh functions, if they are both even or both odd about $\frac{1}{2}$, will be even. If one is odd and one is even, their product will be odd. This is also true with the symmetries about $\frac{1}{4}$, $\frac{1}{8}$, etc. Since this product function is completely described by its symmetry properties about the points 2^{-i}, $i=1,2,3,\ldots$, it must be concluded that this product is also a Walsh function. Moreover, the index of the product is the modulo 2 sum of the indices of the two factors. Thus:

$$W(k,t)\cdot W(m,t) = W(k\oplus m,t)$$

where $\oplus$ indicates addition modulo 2.

$$\begin{aligned} \text{i.e. } W(2,t)\cdot W(3,t) &= W(10_2,t)\cdot W(11_2,t) \\ &= W(10_2\oplus 11_2,t) \\ &= W(01_2,t) \\ &= W(1,t) \end{aligned}$$

Since the reciprocal of a Walsh function is the same function,

$$W(k,t) = 1/W(k,t)$$

the Walsh functions form a closed set under multiplication or division.

A useful property of discrete Walsh functions is that

$$W_N(j,k) = W_N(k,j),$$

that is the matrix representation, Figure 4, is symmetric.

A Hademard matrix is an N x N matrix, each of whose elements are +1 or -1 and such that the rows of the matrix are orthogonal to each other using the ordinary vector inner product. Since the discrete Walsh function representation conforms to this definition, it is a symmetric Hadamard matrix. If H denotes the matrix, then:

$H \cdot H^T = NI$ where H^T is H transpose

N is the order and

I is the identity matrix

but H is symmetric, so $H = H^T$, and therefore, $HH = NI$. This important property will be used in Section IV, The Walsh Transform.

III. GENERATION OF WALSH FUNCTIONS

Walsh functions can be generated in several different ways, all of which depend upon the mathematical properties of the functions as discussed in Section II. Figure 5 shows a hardware realization of a Walsh function generator designed by Harmuth [Ref. 9]. This generates Walsh functions as products of Rademacher functions as discussed in Appendix A.

Swick [Ref.22 and 5] showed that by using the properties of symmetry, a particular Walsh function can be written down directly from left to right without making use of any preceding function. This method is to expand the index of the desired function into binary.

i.e. $k = b_n\ b_{n-1}\ \ldots b_1$, $N = 2^n$,

and let $W(k,t) = 1$ in the interval $[0,1/N]$. Then for $j = n$, $n-1$, ..., 1 if $bj = 1(0)$ then $W(k,t)$ is odd(even) about 2^{-j} when it is considered as a function over the interval $[0,2^{-(j-1)}]$. Figure 6 shows the generation of $W(11,t)$ by this method. Eleven in binary is 1011 and $n = 4$. Letting $W(11,t) = 1$ in the interval $[0,1/16]$, then $b_4 = 1$ indicates an odd function about 1/16 on the interval $[0,1/8]$; $b_3 = 0$ indicates an even function about 1/8 over the interval $[0,1/4]$; $b_2 = 1$ indicates an odd function about 1/4 over $[0,1/2]$; and $b_1 = 1$ indicates an odd function over the entire interval $[0,1]$.

Peterson, [Ref.18], devised a method similar to Swicks in which the index is expanded into reflected binary (Grey code).

i.e. $k = g_n\ g_{n-1} \cdots g_1$, $W(k,t) = 1$ in the interval $[0,1/N]$ and then for $j = n, n-1, \ldots, 1$; $W(k,t)$ in the interval $[2^{-j}, 2^{-(j-1)}]$ is equal to $(-1)^{g_n} \cdot W(k,t)$ in the interval $[0,2^{-j}]$. Referring again to Figure 6 for the generation of $W(11,t)$, and using the algorithm from Appendix A for conversion from binary to reflected binary, 11 in reflected binary is 1110. Letting $W(11,t) = 1$ in the interval $[0,1/16]$, then $(-1)^{g_4} = -1$ and $W(11,t)$ in the interval $[1/16,1/8]$ equals $-W(11,t)$ in the interval $[0,1/16]$. Similarly, $W(11,t)$ in $[1/8,1/4]$ equals $-W(11,t)$ in $[0,1/8]$; $W(11,t)$, $1/4 \ t \ 1/2$ equals $-W(11,t)$ for $0 \ t \ 1/2$; and the function from $t = 1/2$ to $t = 1$ is equal to $W(11,t)$ over $[0,1/2]$.

It should be noted that by using Swick's method but expanding the index in reflected binary or by using Peterson's method with the index expanded in binary, the Walsh functions generated are in binary order instead of sequency order.

Figures 7 and 8 show hardware versions of a fast Walsh transform algorithm which will be discussed in further detail in Section IV. A single pulse input into the device of Figure 7 or a pulse shifted through the input register of Figure 8, will produce as an output the Walsh functions instead of the sequency spectral components.

Computer program 1 utilizes a set of algorithms [Ref.11, 12, and 13] to form a Hadamard matrix of Walsh functions in either binary or sequency order. These algorithms compute the value of the Walsh function at each point without reference to previous points or previous functions. This is a

relatively slow method of function generation since it does not take advantage of the symmetry properties of the Walsh functions.

Computer program 2 also generates a Hadamard matrix in sequency order using an algorithm based on Peterson's method of Walsh function generation.

IV. THE WALSH TRANSFORM

It was shown in Section II that a function can be expressed in a series based on Walsh functions analogous to a Fourier series. In this series, a set of coefficients, $\{a_k\}$, represented the sequency spectrum of the given function. The discrete or sampled form of the Walsh transpose became:

$$a_k = 1/N \sum_{j=0}^{N-1} f(j\tau)\cdot W_N(k,j) \qquad \text{(IV-1)}$$

If the samples of $f(j\tau)$ are arranged in a column matrix,

$$F = \begin{bmatrix} f(0) \\ f(\ \tau) \\ f(2\tau) \\ \cdot \\ \cdot \\ \cdot \\ f[(N-1)\tau] \end{bmatrix}$$

and premultiplied by the Hadamard representation of the Walsh functions, equation (IV-1) is realized in matrix form as:

$$W = 1/N\ HF \qquad \text{(IV-2)}$$

where W is a column matrix of the spectral coefficients. Because of this property, the finite Walsh transform is often called the Hadamard transform.

From the properties of H as described in Section II, it can be readily seen that:

$$HW = 1/N\ HHF = NI/N\ F = F \qquad \text{(IV-3)}$$

or that with the exception of a constant multiplier, 1/N, the same procedure is used to obtain both the transform and the inverse transform.

Some of the properties of the finite Walsh transform [Ref. 15], are as follows:

if $f(j\tau) \longleftrightarrow W(f)$

and $g(j\tau) \longleftrightarrow W(g)$

are transform pairs, then:

(1) The transform is linear

$$a \cdot f(j\tau) + b \cdot g(j\tau) \longleftrightarrow a \cdot W(f) + b \cdot W(g)$$

for real constants a and b.

(2) The coefficient of index 0 is the sample mean of f(j)

$$a(0) = 1/N \sum_{j=0}^{N-1} f(j\tau)$$

Logical or dyadic convolution of two sequences f(j) and g(j) is defined as:

$$h(j\tau) = 1/N \sum_{k=0}^{N-1} f(k\tau) g[(j \oplus k)\tau] \qquad \text{(IV-4)}$$

where $\oplus$ indicates modulo 2 addition

(3)

$$h(j\tau) = f(j) \otimes g(j) \longleftrightarrow W(f) \cdot W(g)$$

where $\otimes$ indicates logical convolution

and (4) $f(j\tau) \otimes f(j\tau) \longleftrightarrow W(f)^2$

This is the analog of the Wiener-Khinchin theorem of the frequency domain. The Walsh power spectrum is the finite Walsh transform of the logical autocorrelation of f(jτ).

Computer program 2 utilizes the Hadamard matrix multiplication method to give both the transform or inverse transform of an input. This method is straight forward but

time requirements can be decreased considerably by using fast transform algorithms much like the fast Fourier transform.

Several approaches exist for fast Walsh transform implementation. One method described by Shanks [Ref. 21] parallels the derivation by Cooley and Tukey for the fast Fourier transform, but even faster methods are available [Ref. 2, 23, and 25] which make use of the internal symmetry properties of the Walsh functions.

A useful method is closely associated with Peterson's method of Walsh function generation. Instead of the reflected binary expansion of the index indicating whether the interval is repeated with or without a sign change, the index indicates whether the interval is added or subtracted with the adjacent interval. For example, for the second spectral component of an eighth order system:

$$f(j\tau) = f_0 f_1 f_2 f_3 f_4 f_5 f_6 f_7$$

$$a_2 = a_{011} \text{ (reflected binary)}$$

$$g_3 = 0, \; g_2 = 1, \; g_1 = 1$$

and $$a_2 = 1/8(f_0 \underset{\overset{\uparrow}{g_3}}{+} f_1) \underset{\overset{\uparrow}{g_2}}{-} (f_2 \underset{\overset{\uparrow}{g_3}}{+} f_3) \underset{\overset{\uparrow}{g_1}}{-} [(f_4 \underset{\overset{\uparrow}{g_3}}{+} f_5) \underset{\overset{\uparrow}{g_2}}{-} (f_6 \underset{\overset{\uparrow}{g_3}}{+} f_7)]$$

Figure 7 depicts a hardware realization of this type system for an eighth order spectrum.

Using this or other fast Walsh transform algorithms reduces the number of operations required to $N\log_2 N$ real additions (and/or subtractions) as compared to $N\log_2 N$ complex

multiplications and $N\log_2 N$ complex additions as required by the fast Fourier transform algorithms. It is therefore much faster to go from the time to the sequency domain than from the time to the frequency domain. Also the inverse Walsh transform can be performed with the same hardware or software as the transform and is therefore much more economical if the cost is calculated in time, computer memory or hardware.

Computer program 3 and Figure 8 depict software and hardware realization of another fast Walsh transform method. In this method, sums and differences of the sample values are formed according to a pattern. The sequency spectrum coefficients are the output and in the hardware version, the sequency corresponding to the coefficient can be found by tracing the most direct path from anyone of the sample input points to the desired output. When an addition is passed through,a zero is written down; when a subtraction is passed through, a one is written down starting at the higher and proceeding to the lower digits. When the output terminal is reached, the digits written down are the reflected binary representation of the index of the coefficient.

The software version, represented by subroutine FASWAL of computer program 3 requires the subroutine function CONVRT to convert the coefficients to the sequency order. These coefficients are in a reversed reflected binary order much as the discrete fast Fourier transform's outputs are in a reversed binary order, [Ref. 8].

To find the corresponding indicies as CONVRT does where

$$W(m) = T(k)$$

W(m) is the Walsh coefficient in sequency order

T(k) is the coefficient in reversed reflected binary order

(1) expand m into binary

$$m = b_n b_{n-1} \cdots b_1$$

(2) convert to reflected binary

$$m = g_n g_{n-1} \cdots g_1$$

(3) reverse order of digits

$$k = g_1 g_2 \cdots g_n$$

(4) use this as the binary representation of k

$$k = \sum_{p=1}^{n} g_p \cdot 2^{n-p}$$

Figure 7 shows another version of this same fast transform algorithm. In this case, the input is applied serially and after N input samples, the output (in parallel) will be the Walsh transform or the inverse transform if the input is the sequency ordered spectral coefficients. Note that as for Figure 8, the output is in reversed reflected binary order.

Either of these realizations have advantages when hardware implemented since the ordering of the output can be predetermined and hard-wired so the reordering is not necessary each time the system is used. Note also that either of these units can be used to generate Walsh functions. The one

in Figure 7 by putting a unit pulse into the input and the one in Figure 8 by shifting a unit pulse down through the input register.

V. DIGITAL FILTER CONSIDERATIONS

One of the traditional methods of digital filtering in the frequency domain is by the use of the convolution theorem. That is, an input function is transformed into the frequency domain using either the Fourier, Laplace, or "Z" transform. This is multiplied by a weighting function, the transform of the impulse response of a desired filter, and the inverse transform is taken, yielding the desired output. The logical convolution theorem does not produce this result.

Much has been written about the transition to and from the Walsh or sequency domain for the purpose of digital filtering but very little material can be found concerning what can be done while in the sequency domain to obtain the desired response from the filter in the time domain.

It is possible to construct a filter with any desired sequency response. There is a rough correlation between sequency and frequency responses of filters but only to the point that the higher sequencies contain more of the higher frequencies than the lower sequencies do. For example, low pass sequency filtering at a sequency cutoff of one-half the highest sequency, the sampling sequency, is equivalent to halving the sampling rate of the input function.

To take advantage of the high speed and other economical properties of the Walsh functions, it would be desirable to be able to obtain a desired frequency response of a digital filter which operates in the sequency domain.

The first step in the investigation of this possibility will be to find the frequency spectrum of the discrete Walsh functions. (The frequency spectrum of the continuous Walsh functions is shown in Appendix B.)

The Z-transform of a Walsh function can be written as:

$$Z[W_N(k,j)] = \sum_{i=0}^{N-1} W_N(k,j)\cdot z^{-i} \qquad \text{(V-1)}$$

noting that for N = 1,

$$Z[W_1(0,0)] = 1 \qquad \text{(V-2)}$$

and for N = 2,

$$Z[W_2(0,j)] = 1+z^{-1}$$

$$Z[W_2(1,j)] = 1-z^{-1} \qquad \text{(V-3)}$$

the following recursive relationship is postulated:

$$Z[W_N(2k+p,j)] = Z[W_{N/2}(k,j)][1+(-1)^{j+p}\cdot z^{N/2}] \qquad \text{(V-4)}$$

$p = 0,1$ $\qquad$ $k = 0,1,2,\ldots,N-1$

$j = 0,1,2,\ldots,N-1$ $\qquad$ $N = 2,4,8,\ldots$

This can be induced by the following argument for N a power of 2, say $N = 2^n$. Expressing k of equation (V-4) in binary as:

$$k = b_n b_{n-1} b_{n-2} \cdots b_1 \qquad \text{(V-5)}$$

The binary representation of k can be converted to reflected binary (see Appendix A) as:

$$\begin{array}{rl} k_2 = & b_n b_{n-1} b_{n-2} \cdots b_1 \\ \oplus & \quad b_n \; b_{n-1} \cdots b_2 \\ \hline k_g = & g_n g_{n-1} g_{n-2} \cdots g_1 \end{array} \qquad \text{(V-6)}$$

Noting that,

$$[2\cdot k]_2 = b_n b_{n-1} \cdots b_1 0 \qquad \text{(V-7)}$$

and

$$[2\cdot k]_g = g_n g_{n-1} \cdots g_1 b_1 \qquad \text{(V-8)}$$

then,

$$[1+2\cdot j]_2 = b_n b_{n-1} \cdots b_1 1 \qquad \text{(V-9)}$$

and

$$[1+2\cdot j]_g = g_n g_{n-1} \cdots g_1 \overline{b_1} \qquad \text{(V-10)}$$

$$\text{where } \overline{b_1} = \begin{cases} 1 \text{ for } b_1 = 0 \\ 0 \text{ for } b_1 = 1 \end{cases}$$

From Peterson's method of Walsh function generation as discussed in Section III, the higher order function is composed of the lower order function over the interval [0,1/2] and $(-1)^{b_1}$ or $(-1)^{\overline{b_1}}$, (for $2\cdot j$ and $2\cdot j+1$, respectively), times the lower order function over the interval [1/2,1]. Since $(-1)^j = (-1)^{b_1}$ and $(-1)^{j+1} = (-1)^{\overline{b_1}}$, equation (V-4) is established.

The Z-transform of a Walsh function will therefore be of the form,

$$Z[W_N(k,j)] = (1\pm z^{-1})(1\pm z^{-2})(1\pm z^{-4})\ldots(1\pm z^{-N/2}) \qquad \text{(V-11)}$$

As can be seen from equations (V-4,7,8,9 and 10), the sequence of signs is determined by the reflected binary code of the index, k; that is,

$$Z[W_N(k,j)] = \prod_n [1+(-1)^r z^{-n}] \qquad \text{(V-12)}$$

$n = 1,2,4,\ldots,N/2$

$r = g_{[\log_2(N/n)-1]}$ where g is the subscripted digit of the reflected binary representation of k.

evaluating,

$$1+z^{-m/2}\Big|_{z=e^{i\omega\tau}} = 1 + e^{-\frac{im\omega\tau}{2}}$$

$i = \sqrt{-1}$

τ = sample period

ω = radian frequency

$$= \frac{2(e^{\frac{im\omega\tau}{4}} \quad e^{-\frac{im\omega\tau}{4}})}{2} \; e^{-\frac{im\omega\tau}{4}}$$

$$= 2\mathrm{Cos}(\frac{m\omega\tau}{4})e^{-\frac{im\omega\tau}{4}} \qquad \text{(V-13)}$$

letting $x = \frac{\omega\tau}{2} = \frac{\pi f}{f_s}$; $f_s = 1/\tau$ = sampling frequency

then,

$$1+z^{-m/2}\Big|_{z=e^{i\omega\tau}} = 2\mathrm{Cos}(\frac{xm}{2})e^{-i\frac{xm}{2}} \qquad \text{(V-14)}$$

Similarly,

$$1-z^{-m/2}\Big|_{z=e^{i\omega\tau}} = 2\mathrm{Sin}(\frac{xm}{2})e^{-i\frac{xm}{2}+\frac{\pi}{2}} \qquad \text{(V-15)}$$

Therefore the frequency spectrum of a discrete Walsh function is simply a product of a group of sine and cosine functions with a linear phase response and can be written down directly upon expansion of the index, k, into reflected binary. For $W_N(k,j)$, expanding k into reflected binary gives,

$$k = g_n g_{n-1} \cdots g_1 \; , \; n = \log_2(N)$$

then the magnitude of the frequency spectral component is given as,

$$|F[W_N(k,j)]| = 2^n|f_n(x)f_{n-1}(2x)\ldots f(2^{n-1}x)| \qquad \text{(V-16)}$$

$$f_p(x) = \begin{cases} \mathrm{Cos}(x) & \text{for } g_p = 0 \\ \mathrm{Sin}(x) & \text{for } g_p = 1 \end{cases}$$

and the phase is given by,

$$\angle F[W_N(k,j)] = -(N-1)x+m\pi/2$$

where

$$m = \sum_{p=1}^{n} g_p$$

For example, the frequency spectrum of $W_{32}(14,j)$ would be,

$$k = 14_{10} = 01110_2 = 01001_g$$

$$N = 32, \ n = 5$$

$$g_5 g_4 g_3 g_2 g_1 = 01001$$

$$|F[W_{32}(14,j)]| = 2^5 |\mathrm{Cos}(x)\mathrm{Sin}(2x)\mathrm{Cos}(4x)\mathrm{Cos}(8x)\mathrm{Sin}(16x)|$$

and the phase is,

$$\angle F[W_{32}(14,j)] = -31x + 2\cdot\frac{\pi}{2} = -31x + \pi$$

Figure 10 shows the amplitude of the frequency spectrum of the first eight Walsh functions of order 16 plotted on the interval $[0 \leq f/f_s \leq .5]$. Note that 0.5 f/f_s is the maximum or Nyquist frequency for digital filter applications. In general it is only necessary to plot the first N/2 spectrums, since by symmetry, the N/2 remaining spectrums, (Figure 11), can be obtained by reversing the f/f_s axis of the lower order frequency spectrums.

One method often used for digital filtering in the frequency domain is to use digital resonators in the form of comb filters with zeros introduced to give the desired spectrum response. A Walsh function resonator can be defined as a digital circuit whose pulse response is a Walsh function. A Walsh resonator can be constructed as shown in Figure 12. Here, the sign on each input to the N input summer is determined by the sign of the particular Walsh function at that point, j. A $W_{32}(14,j)$ resonator is shown in Figure 13.

The z-transform of $W_N(k,j)$, equation (V-11), leads to another and simpler realization of the Walsh resonator. This method, shown in Figure 14, uses cascaded sections of lower order filters. The $W_{32}(14,j)$ resonator of this type is shown in Figure 15. This is the method which led to the development of the fast Walsh transform algorithm discussed in Section IV. The hardware realization of that algorithm shown in Figure 7 is simply a full set of N Walsh resonators.

Referring again to Figures 10 and 11, it can be seen that various frequency responses can be obtained by cascading and paralleling various combinations of Walsh resonators. There is no method known that will give the classic filter frequency responses such as Butterworth, Chebychev, or elliptic using this method. A function minimization of the mean square error between a desired frequency response and the frequency response of cascaded/parallel Walsh resonators can give the necessary combination whose frequency response is as close to the desired response as is possible with a given number of resonators. This method will lead to results comparable to that of using a digital comb filter and resonators in the frequency domain, [Ref. 8].

As an example, a filter is desired with a narrow bandwidth at approximately $f/f_s = .25$, using three Walsh resonators. Using the $W_{16}(7,k)$ resonator as a starting approximation, it can be seen that the bandwidth can be decreased and sidelobe magnitudes decreased by cascading it with notch filters constructed from $W_{16}(6,k)$ and $W_{16}(9,k)$ resonators.

The notch filters are simply resonators whose output is subtracted from the input in a feed forward manner. This is made possible by the linear phase characteristics of the Walsh resonators.

Figure 16 shows a block diagram of such a filter and Figure 17 shows the frequency response of the filter. The 3db bandwidth is about $.04f/f_s$ and the nearest sidelobe is down approximately 16db from the main lobe. Neither of the diagrams takes into account the delay errors which would probably be introduced by such a system but these errors can be eliminated much in the same way as they are in frequency domain digital filters. Filters with higher Q's and better sidelobe suppression can be obtained by using higher order Walsh resonators and/or more of them in the system.

VI. RESULTS AND CONCLUSIONS

Many computer runs of both an on-line and off-line nature using the Walsh transforms for digital filtering have been made with various types of inputs and filter responses. After this work, it was concluded that the use of Walsh functions is not the absolute answer to digital filtering or to the many other fields of application being investigated. However, they do appear to have definite applications to assist in overcoming certain problems.

The areas, other than multiplexing [Ref. 4, 14, 15 and 17], where greater benefits may be obtained as compared to the conventional methods appear to be where either the input or output function is of a discontinuous nature. One area where this has been put to use is in the field of image processing [Ref. 1, 3 and 19]. Here the input function, normally two dimensional, may have noise bursts included in it which may be attenuated by low pass sequency filtering or may have low contrast areas which can be enhanced by high pass sequency filtering. In the first case, the input has discontinuities, the noise bursts; in the second, the output requires the presence of discontinuities, the image edges. It has been shown that digital image processing by multi-dimensional Walsh methods gives essentially equivalent results to Fourier methods but with the speed increase inherent in the sequency domain.

Image processing and the field of pattern recognition [Ref. 6] have much in common and again if the pattern to be processed is of a discontinuous form, it is believed that Walsh functions may again be superior to Fourier methods.

Another signal which has characteristics that lend themselves to Walsh functions is one of the "infinitely" clipped type. Infinitely clipped signals also only take on two values and can be represented by Walsh functions easily. The use of Walsh functions in this area, especially for application to infinitely clipped speech processors and vocoders should be investigated further.

Walsh functions will probably gain even greater status in the fields of image processing and multiplexing but with the exception of very specialized applications, their use to filter sinusoids will probably remain quite limited.

APPENDIX A

Walsh Functions as Products of Rademacher Products

The Rademacher functions can be defined as follows:

$$R(0,t) = \begin{cases} 1, & 0 \leq t \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

$$R(1,t) = \begin{cases} 1, & 0 \leq t \leq \frac{1}{2} \\ -1, & \frac{1}{2} \leq t \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

then

$$R(n,t) = R(n-1,2t) + R(n-1,2t-1) \qquad \text{(A-1)}$$

$$n = 2,3,4,\ldots.$$

Each Rademacher function is a replica of the preceding function with the sequency doubled. Referring to Figure 2 and Figure 3, the first 5 Rademacher functions are $W(0,t)$, $W(1,t)$, $W(3,t)$, $W(7,t)$, and $W(15,t)$. That is, the Rademacher functions are the Walsh functions numbered 2^n-1, $n = 0,1,2,\ldots.$ Defining a different method of ordering the Walsh functions, $W(m,t)_b$ and expressing m in binary form,

$$m = g_n g_{n-1} \cdots g_2 g_1$$

Then:

$$W(m,t)_b = \prod_{j=1}^{n} R(j,t)^{g_j} \qquad \text{(A-2)}$$

i.e.

$$W(4,t)_b = W(100_2,t)_b = R(3,t)$$

$$W(5,t)_b = W(101_2,t)_b = R(3,t) \cdot R(1,t)$$

Noting that $W(4,t)_b = W(7,t)$ and $W(5,t)_b = W(6,t)$, the functions $W(m,t)_b$ are not in the same order as $W(k,t)$. To find the index, m, in $W(m,t)_b$ which corresponds to the index, k, in $W(k,t)$, it is necessary to represent $W(k,t)$ as the product of Rademacher functions, remembering that $R(i,t) = W(2^i-1,t)$. From the product rule, $W(p,t)\cdot W(q,t) = W(p\oplus q,t)$ and expressing k in its binary digits: $k = b_n b_{n-1} b_{n-2} \ldots b_1$ the desired relationship can be found.

From (A-2):

$$W(m,t)_b = \prod_{j=1}^{n} R(j,t)^{g_j} = \prod_{j=1}^{n} W(2^j-1,t)^{g_j}$$

but 2^{j-1} in binary is of the form $111\cdots\cdot 1$ and k must be the modulo 2 sum of several factors of this type. This can be expressed as:

$$b_n b_{n-1} \ldots b_1 = \underbrace{g_n g_n g_n \ldots g_n}_{n \text{ digits}} \oplus \underbrace{g_{n-1} g_{n-1} \ldots g_{n-1}}_{n-1 \text{ digits}} \oplus \ldots \oplus g_1$$

Adding the right side gives,

$$b_n b_{n-1} \ldots b_1 = g_n (g_n \oplus g_{n-1})(g_n \oplus g_{n-1} \oplus g_{n-2}) \ldots (g_n \oplus \ldots \oplus g_1)$$

or,

$$b_j = g_n \oplus g_{n-1} \oplus \ldots \oplus g_{j+1} \oplus g_j .$$

Therefore

$$g_n = b_n$$

$$g_n \oplus g_{n-1} = b_n \oplus g_{n-1} = b_{n-1}$$

or

$$g_{n-1} = b_n \oplus b_{n-1} .$$

and

$$g_j = b_j \oplus b_{j-1}, \quad b_{n+1} \equiv 0$$

Thus the binary representation of m, $g_n g_{n-1} \cdots g_1$, is the reflected binary (Grey code) representation of k_{binary}. That is,

$$W(b_n b_{n-1} \cdots b_1, t) = W(g_n g_{n-1} \cdots g_1, t)_b \qquad \text{(A-3)}$$

The following algorithms can be used to convert from binary to reflected binary or vice versa.

$$g_j = b_j \oplus b_{j-1}; \quad b_j = g_j \oplus b_{j+1}; \quad b_{n+1} \equiv 0 \qquad \text{(A-4)}$$

This is illustrated by the following diagram.

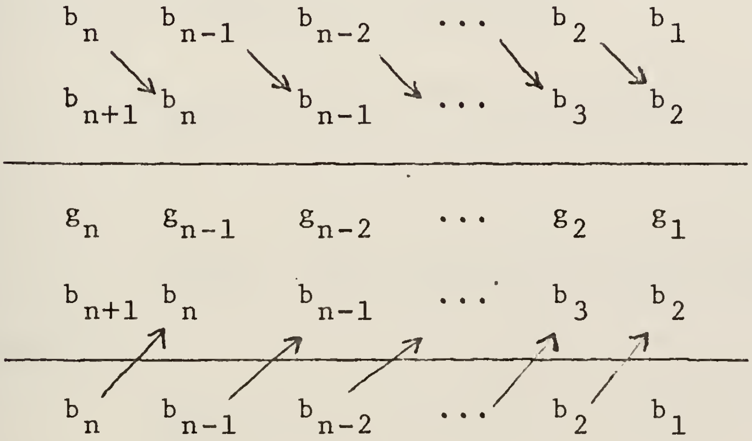

b_n b_{n-1} b_{n-2} ... b_2 b_1
b_{n+1} b_n b_{n-1} ... b_3 b_2
g_n g_{n-1} g_{n-2} ... g_2 g_1
b_{n+1} b_n b_{n-1} ... b_3 b_2
b_n b_{n-1} b_{n-2} ... b_2 b_1

Figure 9 shows a set of discrete Walsh functions generated as the product of Rademacher functions and left in binary order.

APPENDIX B

Frequency Spectrum (Fourier Transform) of the Continuous Walsh Functions

The Fourier transform, $F(w)$, of a function is defined as,

$$F(w) = \int_{-\infty}^{\infty} e^{-iwt} f(t)\, dt \qquad i = \sqrt{-1}$$

w = radian frequency

therefore,

$$F_k(w) = \int_0^1 e^{-iwt} W(k,t)\, dt$$

and by the definition of $W(k,t)$,

$$F_{2k}(w) = \int_0^1 e^{-iwt} W(k,2t)\, dt + (-1)^k \int_0^1 e^{-iwt} W(k,2t-1)\, dt$$

or

$$F_{2k}(w) = 1/2\ F_k(w/2)[1+(-1)^k e^{-\frac{iw}{2}}]$$

Similarly,

$$F_{2k\ 1}(w) = 1/2\ F_k(w/2)[1-(-1)^k e^{-\frac{iw}{2}}]$$

Therefore,

$$F_0(w) = \frac{1}{iw}(1-e^{-iw}) = \frac{2e^{-\frac{iw}{2}}}{w}\ \mathrm{Sin}(w/2)$$

$$F_1(w) = 1/2\left[\frac{2e^{-\frac{iw}{4}}}{w/2}\ \mathrm{Sin}(w/4)\right](1-e^{-\frac{iw}{2}})$$

$$= \frac{4i}{w}\ e^{-\frac{iw}{2}}\ \mathrm{Sin}^2(w/4)$$

$$F_2(w) = \frac{8(i)^2}{w} e^{-\frac{iw}{2}}\ \mathrm{Sin}^2(w/8)\ \mathrm{Sin}(w/4)$$

$$F_3(w) = \frac{8i}{w}\ e^{-\frac{iw}{2}}\ \mathrm{Sin}^2(w/8)\ \mathrm{Cos}(w/4)$$

By induction, a general form of $F_k(w)$ can be established as follows:

expanding k in reflected binary,

$$k = g_n g_{n-1} \cdots g_1$$

let

$$m = \sum_{j=1}^{n} g_j$$

then,

$$f_k(w) = \frac{2^{n+1}(i)^m}{w} \cdot e^{-\frac{iw}{2}} \cdot \text{Sin}^2(w/2^{n+1}) \cdot$$

$$\prod_{j=1}^{n-1} \text{Sin}^{g_j}(w/2^{j+1}) \cdot \text{Cos}^{(1-g_j)}(w/2^{j+1})$$

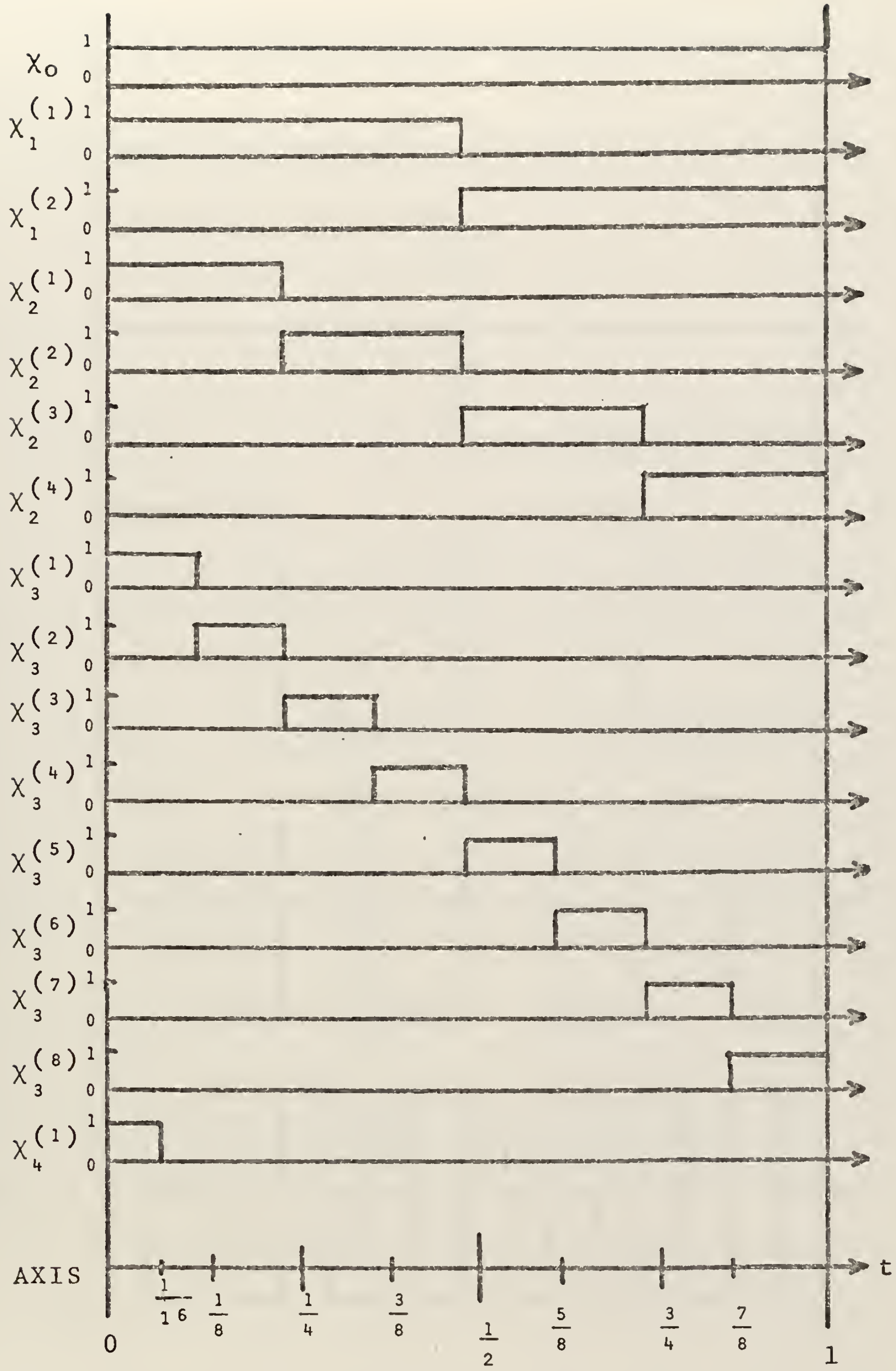


Fig. 1. The first 16 Harr Functions, $\chi_k^{(i)}(t)$.

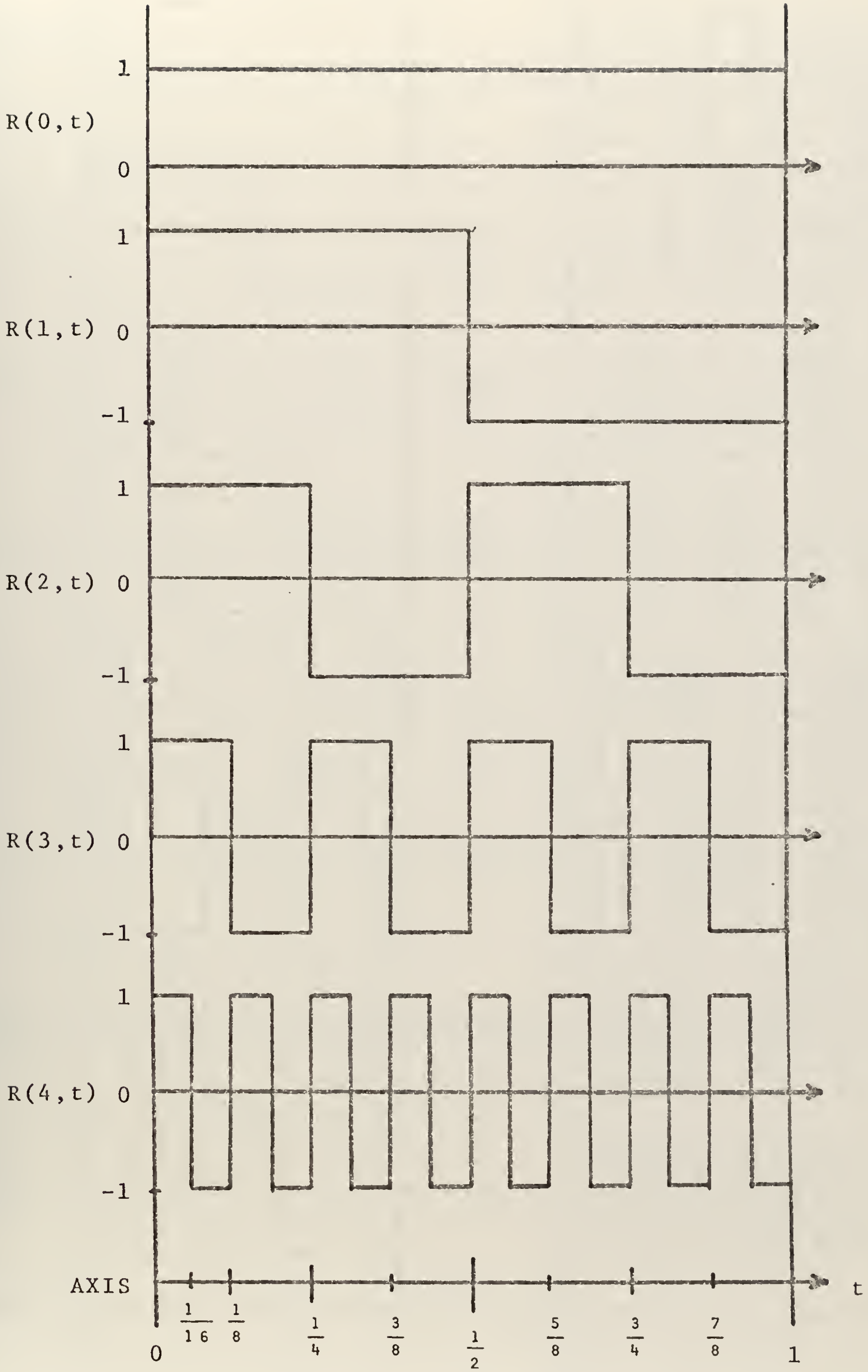


Fig. 2. The First Five Rademacher Functions, R(k,t).

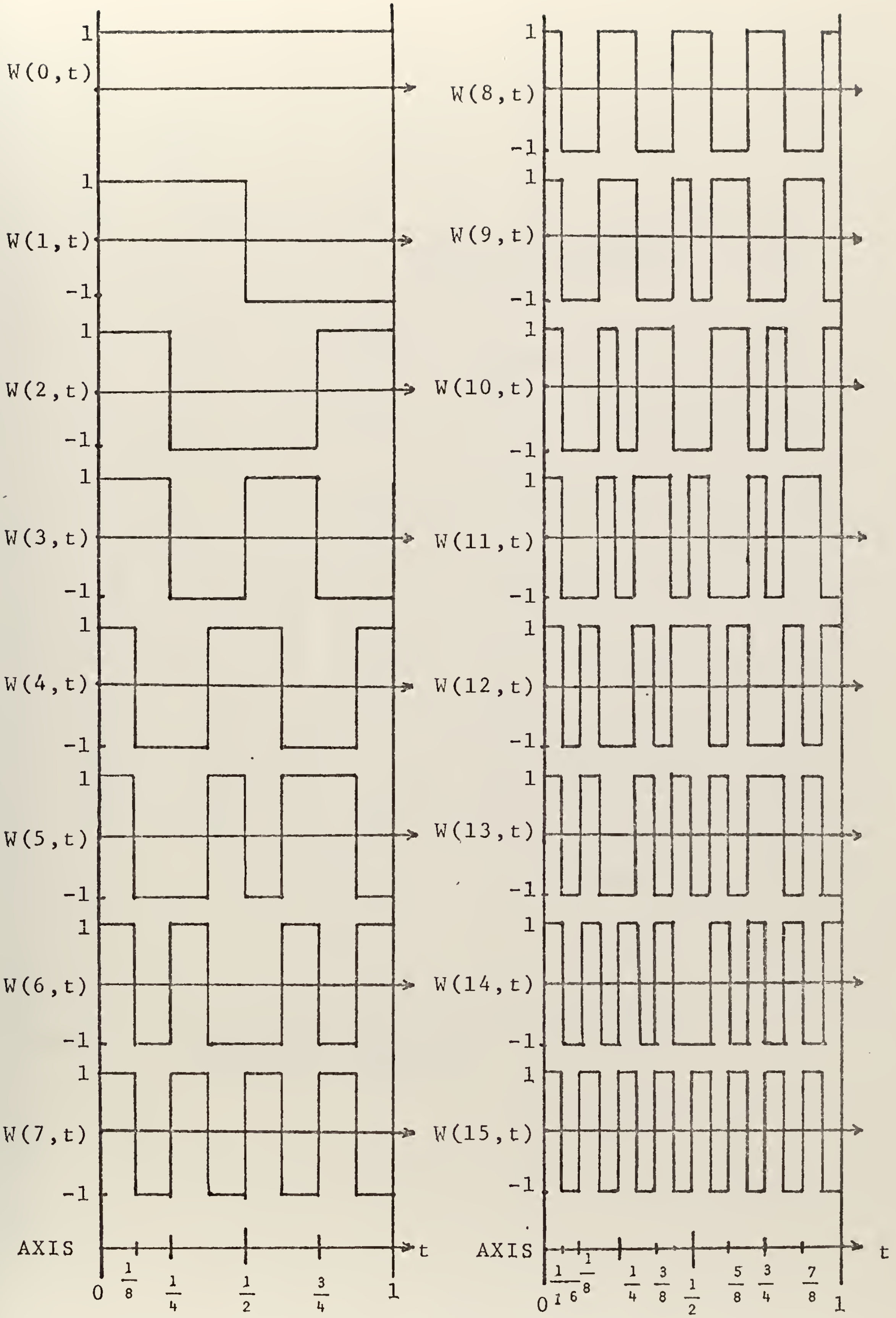


Fig. 3. The First 16 Walsh Functions, W(k,t).

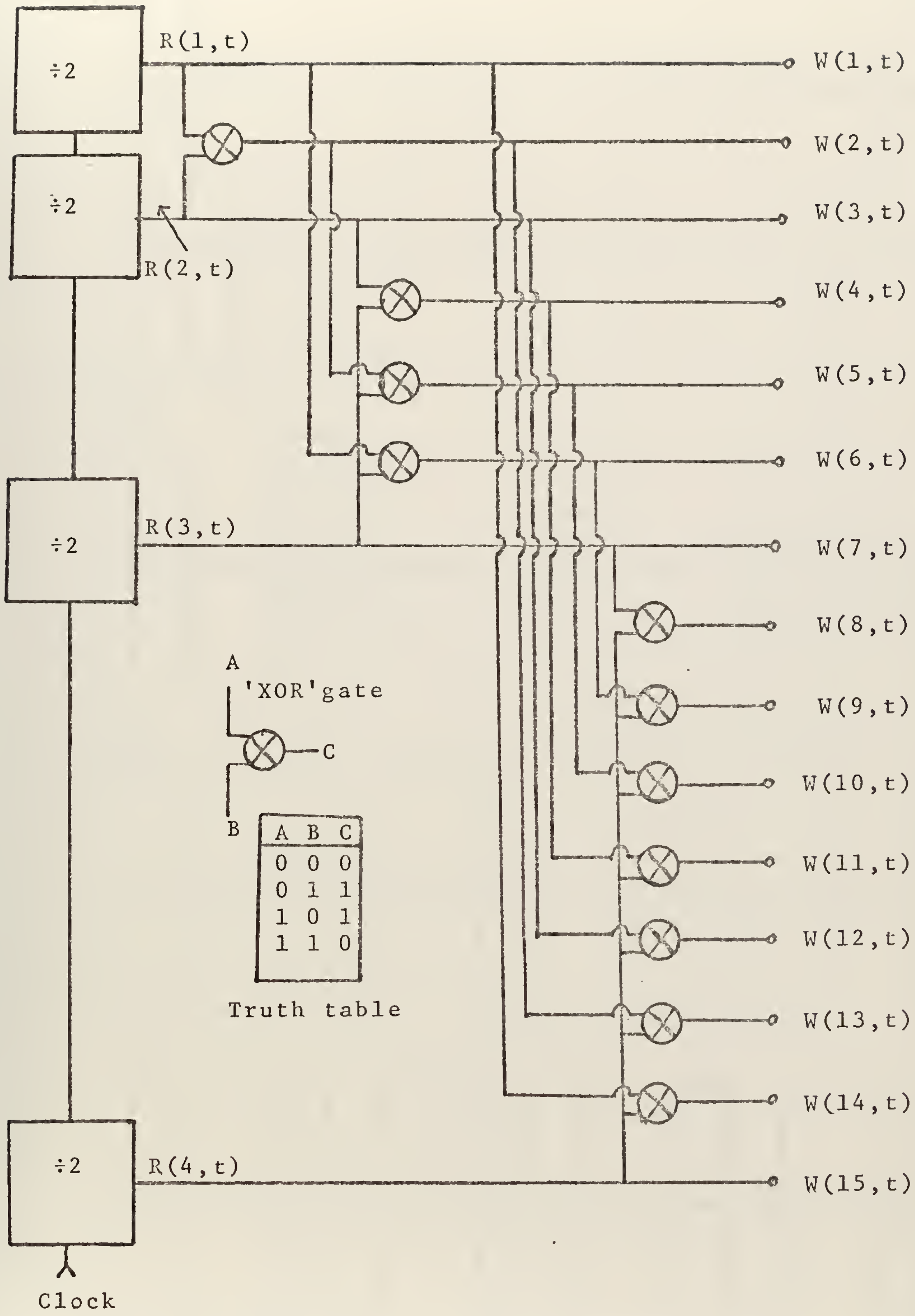
```
                 H A D A M A R D   M A T R I X
                      (SEQUENCY ORDERED)

                          ORDER= 16

                            W(K,J)=
                            16

 J=   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
  K=
  0   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
  1   1   1   1   1   1   1   1   1  -1  -1  -1  -1  -1  -1  -1  -1
  2   1   1   1   1  -1  -1  -1  -1  -1  -1  -1  -1   1   1   1   1
  3   1   1   1   1  -1  -1  -1  -1   1   1   1   1  -1  -1  -1  -1
  4   1   1  -1  -1  -1  -1   1   1   1   1  -1  -1  -1  -1   1   1
  5   1   1  -1  -1  -1  -1   1   1  -1  -1   1   1   1   1  -1  -1
  6   1   1  -1  -1   1   1  -1  -1  -1  -1   1   1  -1  -1   1   1
  7   1   1  -1  -1   1   1  -1  -1   1   1  -1  -1   1   1  -1  -1
  8   1  -1  -1   1   1  -1  -1   1   1  -1  -1   1   1  -1  -1   1
  9   1  -1  -1   1   1  -1  -1   1  -1   1   1  -1  -1   1   1  -1
 10   1  -1  -1   1  -1   1   1  -1  -1   1   1  -1   1  -1  -1   1
 11   1  -1  -1   1  -1   1   1  -1   1  -1  -1   1  -1   1   1  -1
 12   1  -1   1  -1  -1   1  -1   1   1  -1   1  -1  -1   1  -1   1
 13   1  -1   1  -1  -1   1  -1   1  -1   1  -1   1   1  -1   1  -1
 14   1  -1   1  -1   1  -1   1  -1  -1   1  -1   1  -1   1  -1   1
 15   1  -1   1  -1   1  -1   1  -1   1  -1   1  -1   1  -1   1  -1
  K=
 J=   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
```

Fig. 4. Discrete Walsh Functions of Order 16
(Sequency Ordered)



A	B	C
0	0	0
0	1	1
1	0	1
1	1	0

Truth table

Fig. 5. Walsh Function Generator(as Rademacher Function Products).

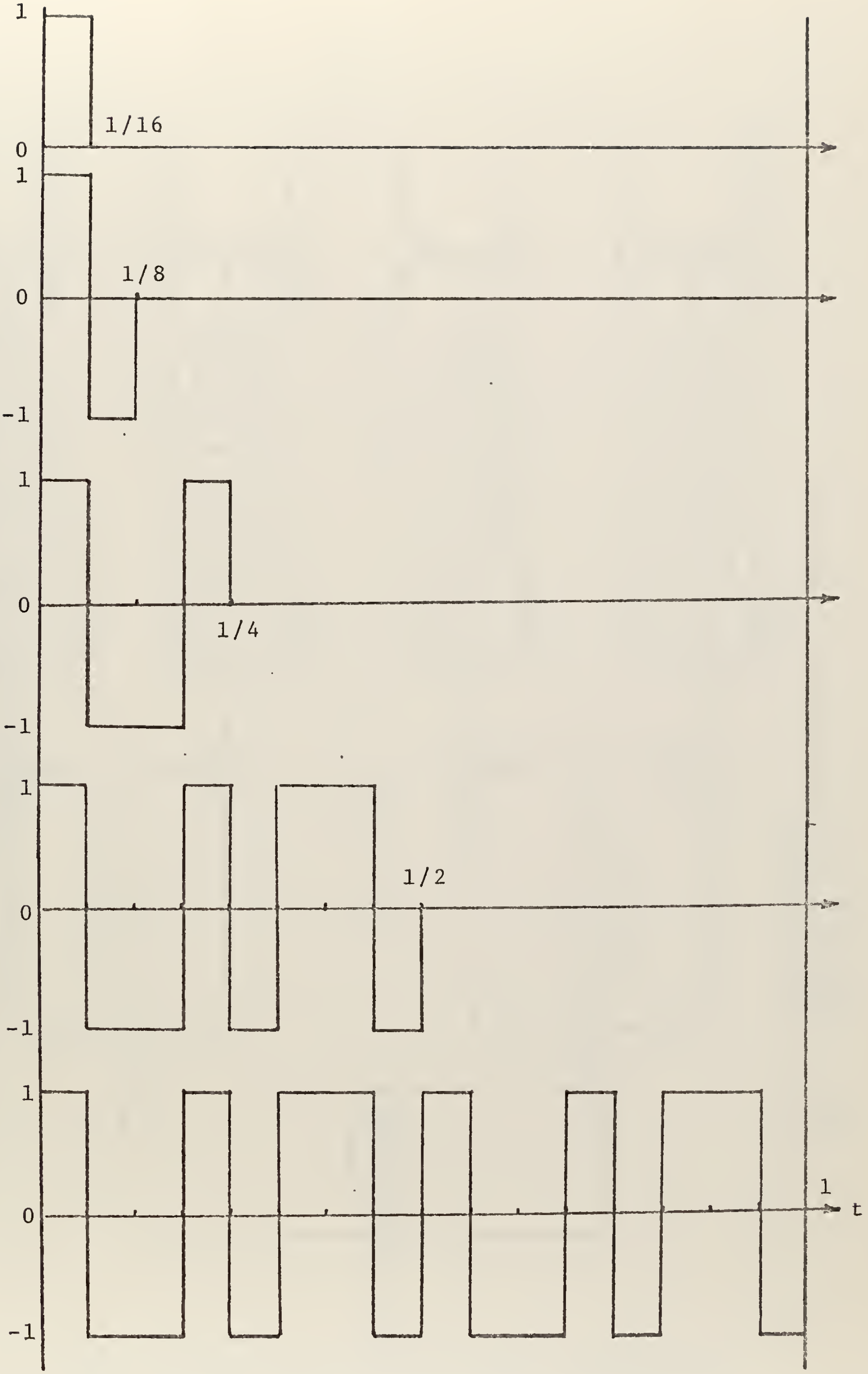


Fig. 6. Generation of $W_{16}(11,t)$.

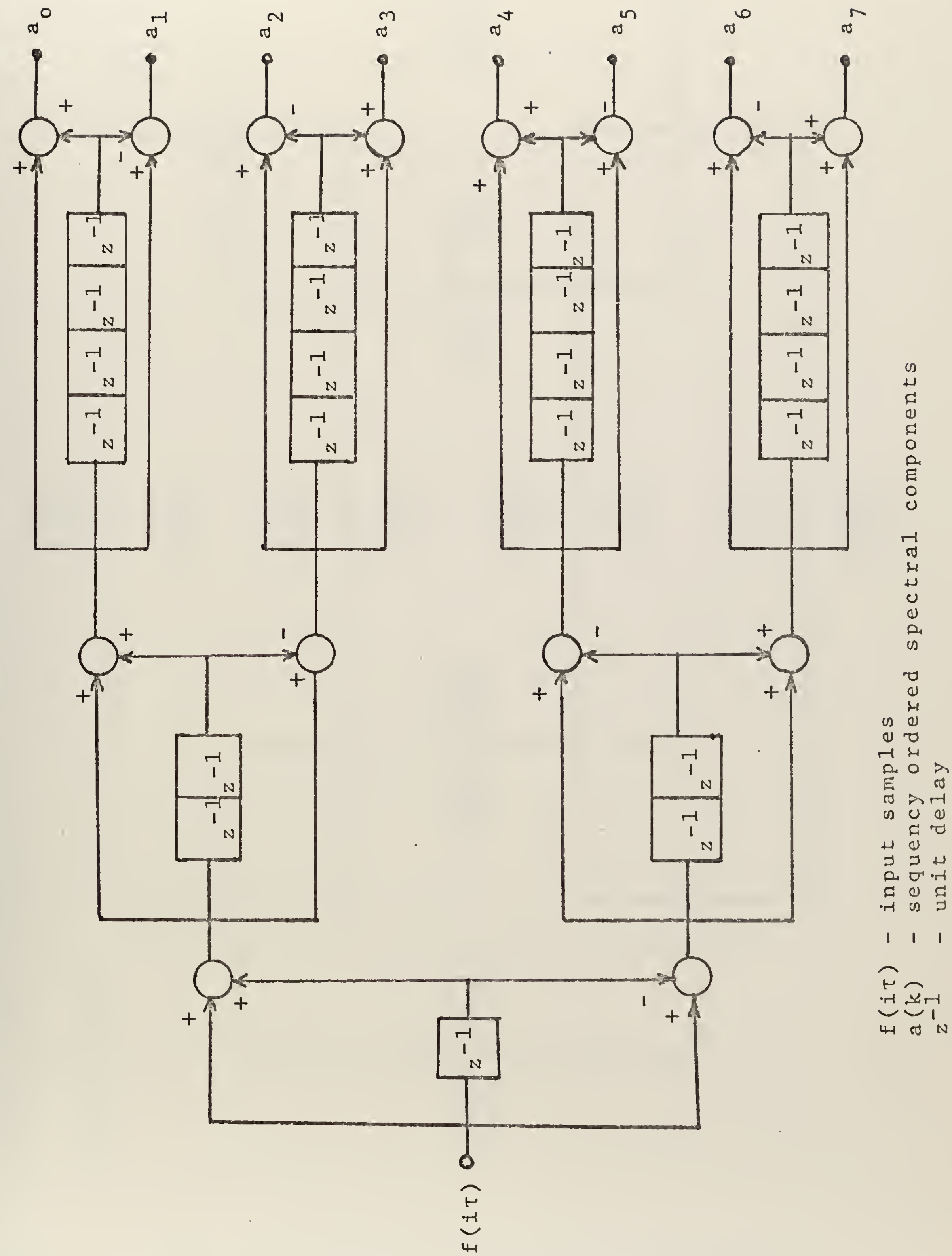


Fig. 7. Serial In/Parallel out Fast Walsh Transform Realization.

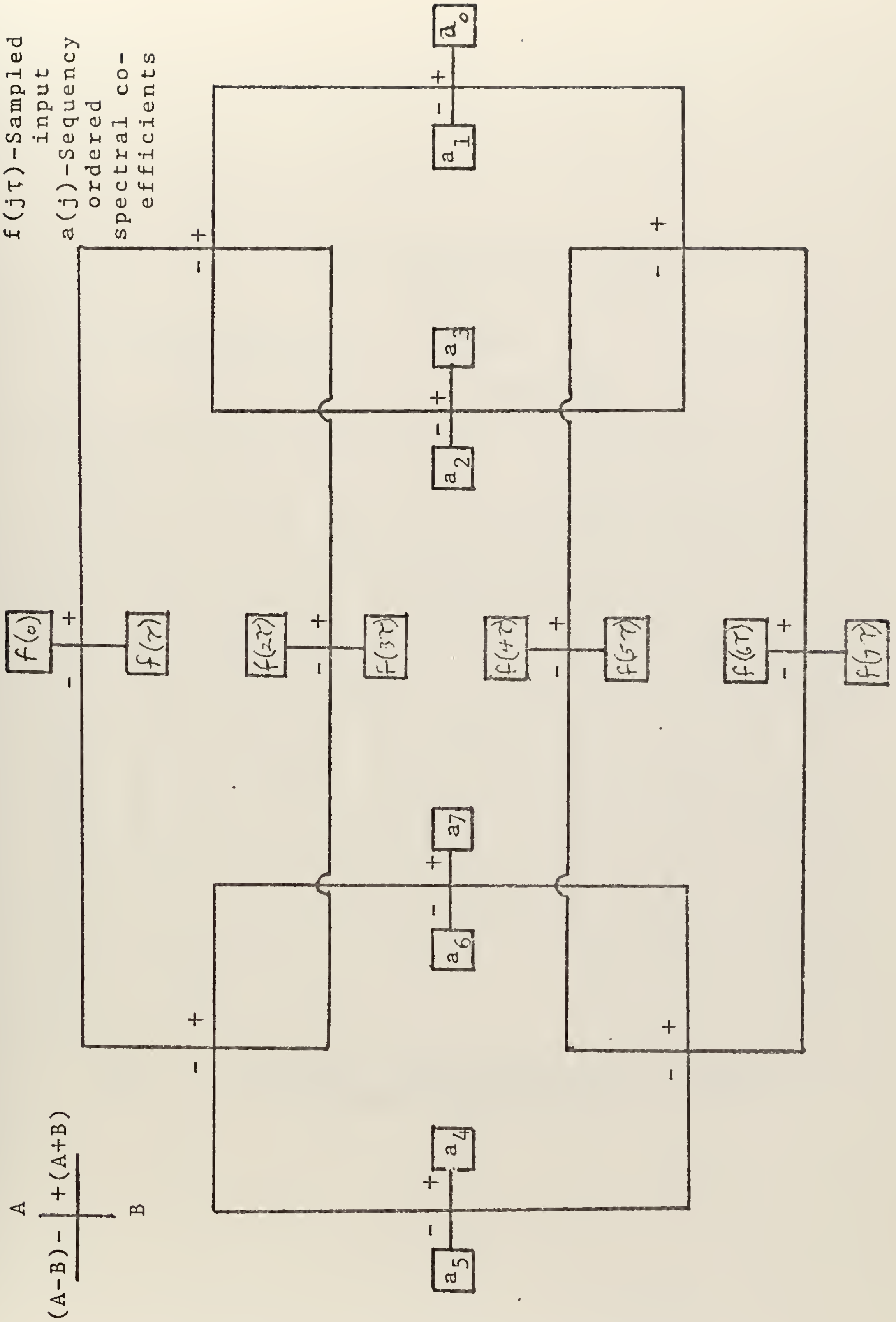


Fig. 8. Fast Walsh Transform Realization Parallel in/ Parallel out

HADAMARD MATRIX
(BINARY ORDERED)

ORDER= 16

$W_{16}(K,J)_b=$

```
J=   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
 K=
 0   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
 1   1   1   1   1   1   1   1   1  -1  -1  -1  -1  -1  -1  -1  -1
 2   1   1   1   1  -1  -1  -1  -1   1   1   1   1  -1  -1  -1  -1
 3   1   1   1   1  -1  -1  -1  -1  -1  -1  -1  -1   1   1   1   1
 4   1   1  -1  -1   1   1  -1  -1   1   1  -1  -1   1   1  -1  -1
 5   1   1  -1  -1   1   1  -1  -1  -1  -1   1   1  -1  -1   1   1
 6   1   1  -1  -1  -1  -1   1   1   1   1  -1  -1  -1  -1   1   1
 7   1   1  -1  -1  -1  -1   1   1  -1  -1   1   1   1   1  -1  -1
 8   1  -1   1  -1   1  -1   1  -1   1  -1   1  -1   1  -1   1  -1
 9   1  -1   1  -1   1  -1   1  -1  -1   1  -1   1  -1   1  -1   1
10   1  -1   1  -1  -1   1  -1   1   1  -1   1  -1  -1   1  -1   1
11   1  -1   1  -1  -1   1  -1   1  -1   1  -1   1   1  -1   1  -1
12   1  -1  -1   1   1  -1  -1   1   1  -1  -1   1   1  -1  -1   1
13   1  -1  -1   1   1  -1  -1   1  -1   1   1  -1  -1   1   1  -1
14   1  -1  -1   1  -1   1   1  -1   1  -1  -1   1  -1   1   1  -1
15   1  -1  -1   1  -1   1   1  -1  -1   1   1  -1   1  -1  -1   1
 K=
J=   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
```

Fig. 9. Discrete Walsh Functions of Order 16
(Binary Ordered)

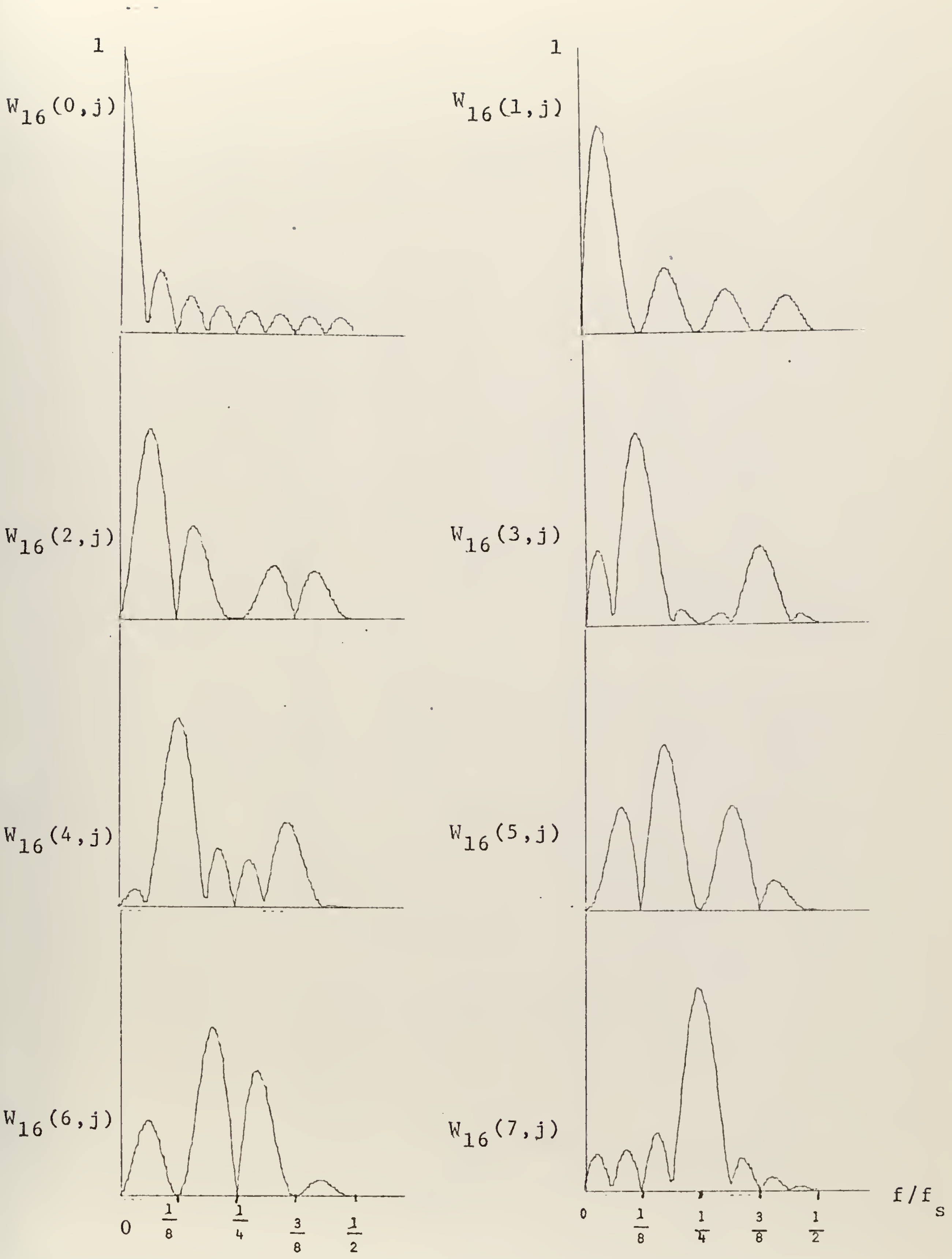


Fig. 10. Frequency Spectrum of First 8 Discrete Walsh Functions, $W_{16}(k,j)$.

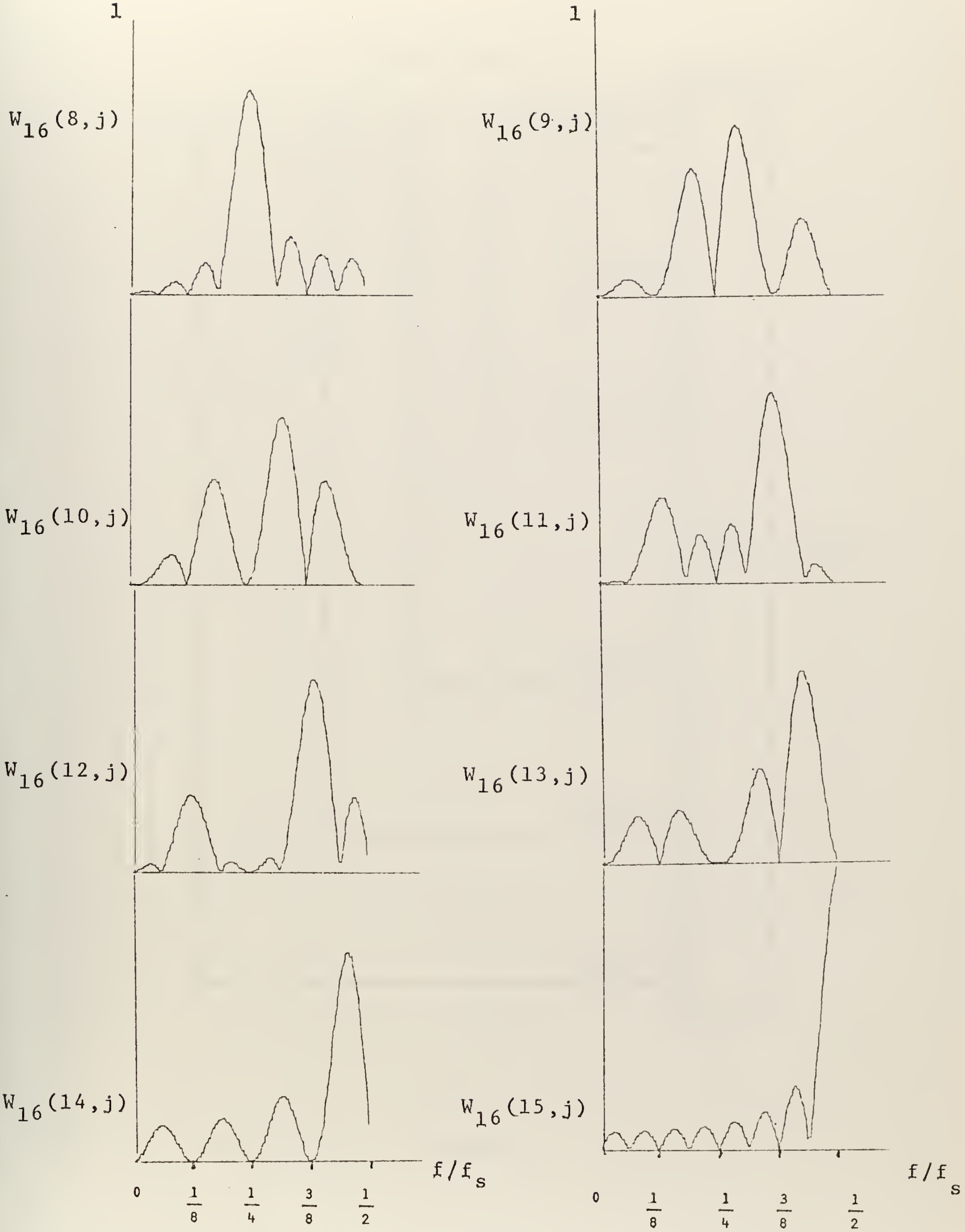


Fig. 11. Frequency Spectrum of Second 8 Discrete Walsh Functions, $W_{16}(k,j)$.

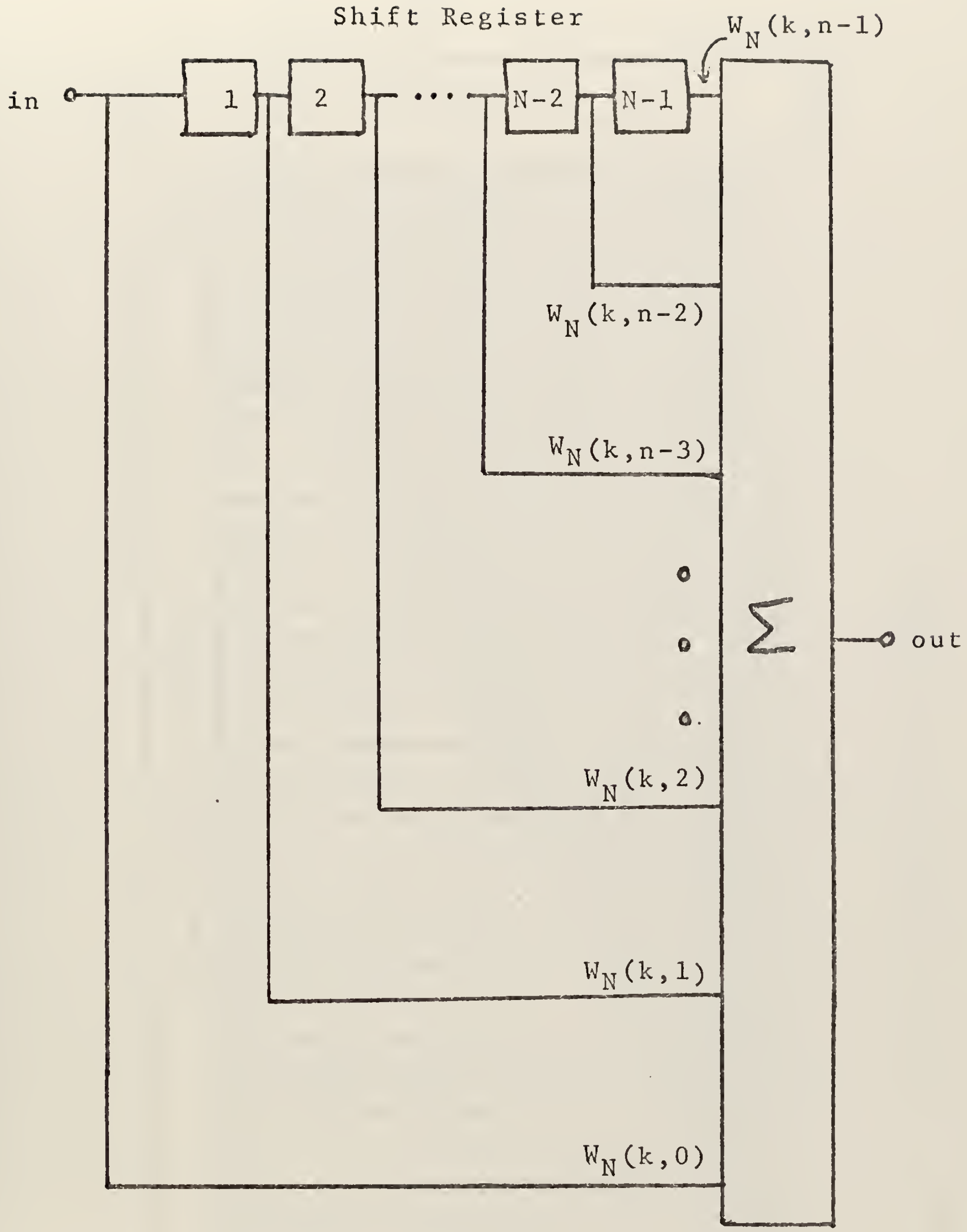


Fig. 12. Walsh Function Resonator.

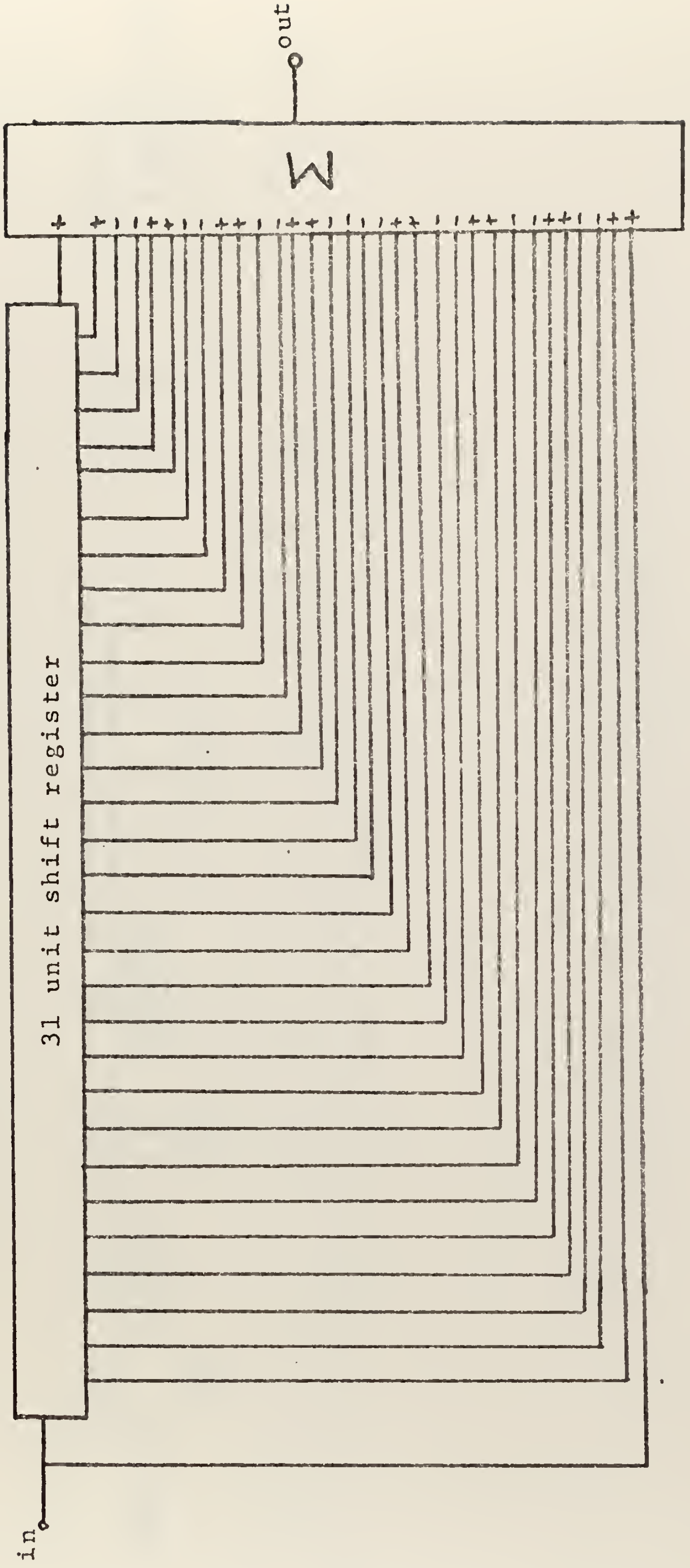
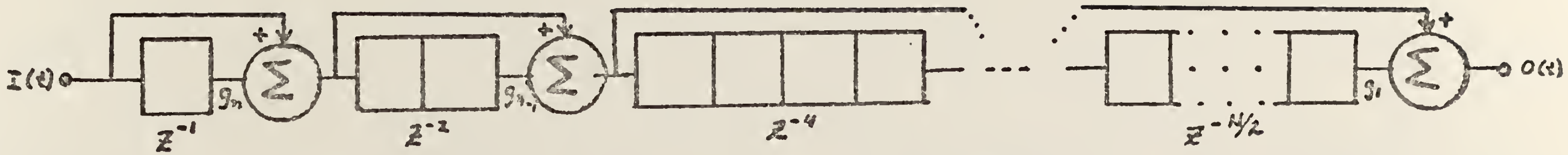


Fig. 13. Walsh Resonator for $W_{32}(14,j)$.



$$O(z) = I(z)[1+(-1)^{g_n} z^{-1}][1+(-1)^{g_{n-1}} z^{-2}]\cdots[1+(-1)^{g_1} z^{-N/2}]$$

Fig. 14. Cascade Version of Walsh Resonator.

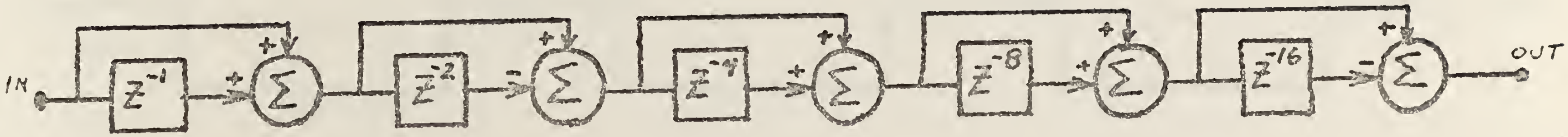


Fig. 15. Cascade Version of $W_{32}(14,j)$ Walsh Resonator.

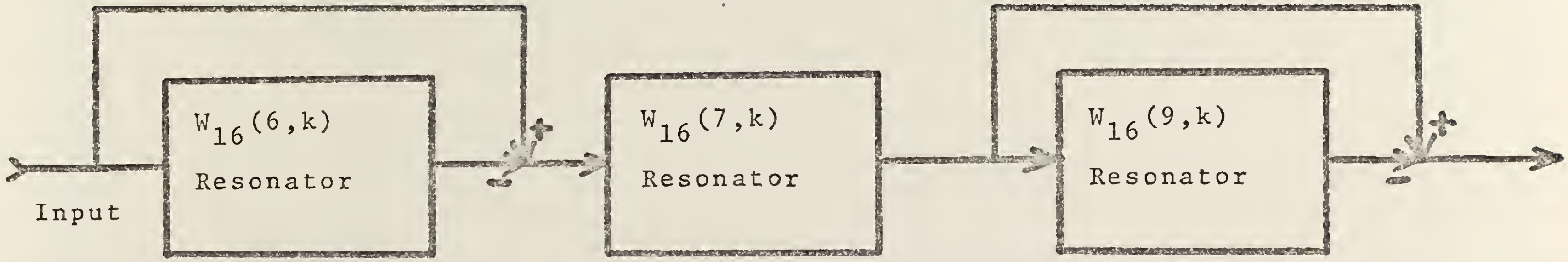


Fig. 16. Narrow Bandpass Filter Using Walsh Resonators.

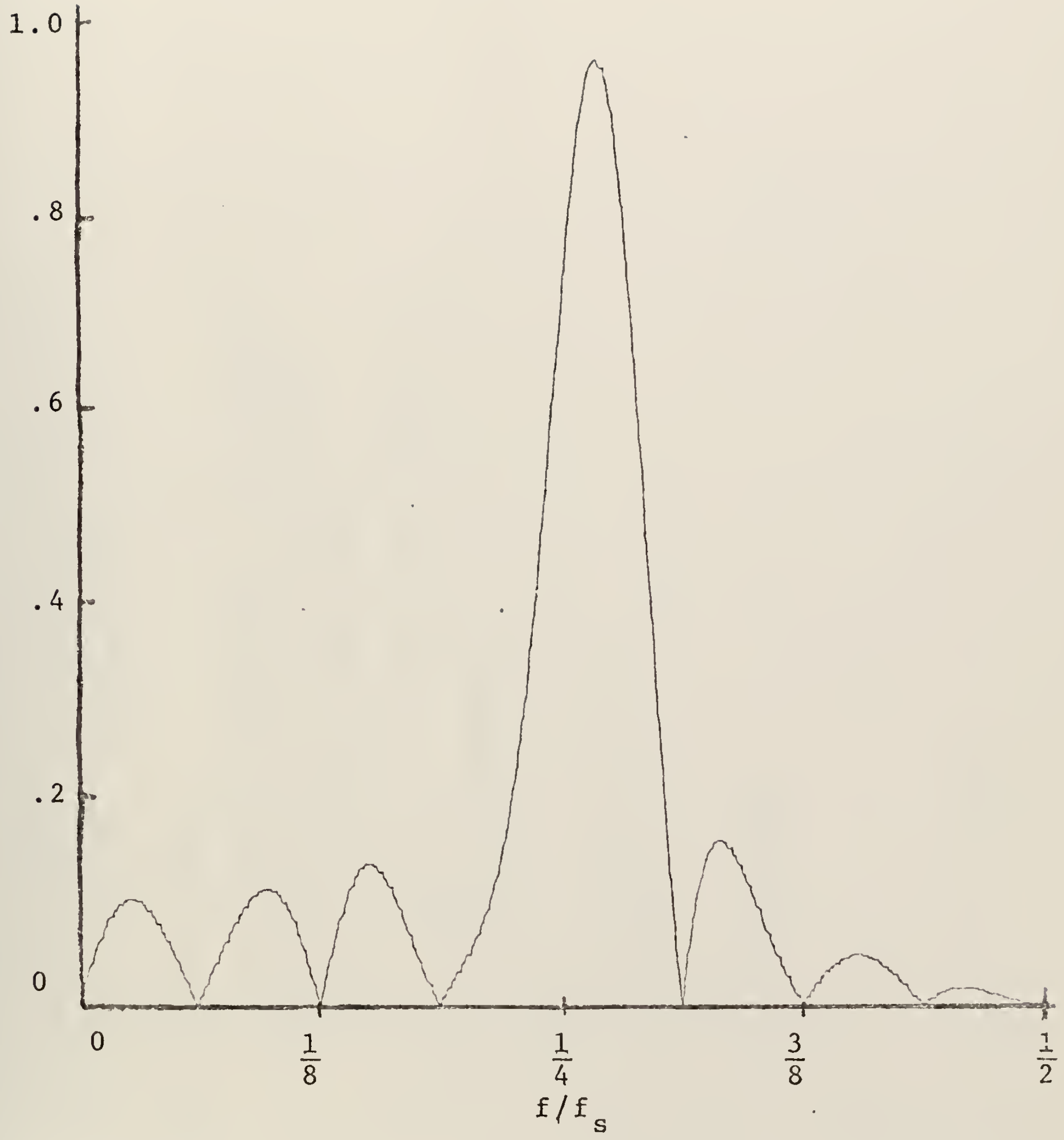


Fig. 17. Frequency Response of Narrow Bandpass Filter Using Walsh Resonators.

COMPUTER PROGRAM 1

Hadamard Matrix Generation

```
C      *************************************************
C      *
C      *  THIS PRØGRAM GENERATES A SET ØF DISCRETE WALSH
C      *  FUNCTIØNS ØF ØRDER N, N=2,4,8,16, ØR 32, AS
C      *  PRØDUCTS ØF RADEMACHER FUNCTIØNS.  THE ØUTPUT
C      *  IS IN THE FØRM ØF A HADAMARD MATRIX.
C      *
C      *  USER INSTRUCTIØNS:
C      *
C      *       ØNE DATA CARD IS REQUIRED.  THE VALUE ØF
C      *       N IS ENTERED IN CØLS. 1+2 IN I2 FØRMAT.
C      *       CØLS. 10-12 CØNTAIN SEQ FØR SEQUENCY
C      *       ØRDER ØR BIN FØR BINARY ØRDER.
C      *
C      *  PRØGRAM LANGUAGE IS IBM-360 SYSTEM FØRTRAN IV
C      *
C      *************************************************
C
C      MAIN PRØGRAM
C
       INTEGER SEQWAL,W(32,32),BIWAL
       DATA SEQ,BIN/3HSEQ,3HBIN/
       READ(5,100)N,TYPE
       WRITE(6,101)
       IF(TYPE.EQ.SEQ)GØTØ10
       IF(TYPE.NE.BIN)GØTØ60
       WRITE(6,102)
       GØTØ20
10     WRITE(6,103)
20     WRITE(6,104)N
       NX=N-1
       WRITE(6,105)((J),J=0,NX)
       WRITE(6,106)
       DØ 40 K=1,N
       KX=K-1
```

```
      DO 40 J=1,N
      A=FLOAT(J-1)/N
      IF(TYPE.EQ.SEQ)GOTO30
      W(K,J)=BIWAL(KX,A)
      GOTO40
30    W(K,J)=SEQWAL(KX,A)
40    CONTINUE
      DO50 K=1,N
      KX=K-1
50    WRITE(6,106)KX,(W(K,J),J=1,N)
      WRITE(6,108)
      WRITE(6,105)((J),J=0,NX)
      STOP
60    WRITE(6,107)
      STOP
100   FORMAT(I2,7X,A3)
101   FORMAT(1H1,4X,'H A D A M A R D   M A T R I X')
102   FORMAT(10X,'(BINARY ORDERED)')
103   FORMAT(10X,'(SEQUENCY ORDERED)')
104   FORMAT(/,12X,'ORDER=',I3,//,15X,'W(K,J)=',/)
105   FORMAT(' J=',32I3)
106   FORMAT(33I3)
107   FORMAT(' ERROR, EXECUTION TERMINATED')
108   FORMAT('  K=')
      END
```

```
      INTEGER FUNCTION SEQWAL(K,X)
C
C     IF SEQUENCY ORDERED FUNCTIONS ARE DESIRED, SEQWAL
C     CONVERTS THE SEQUENCY ORDERED INDEX TO THE CORRECT
C     BINARY ORDERED INDEX.
C
      INTEGER BIWAL
      K2=K/2
      L=K
      M=1
      N=0
      I=0
10    I=I+1
      I1=0
      I2=0
      M2=M*2
      IF(MOD(K,M2)) 20,20,15
15    K=K-M
      I1=1
20    IF(MOD(K2,M2)) 30,30,25
25    K2=K2-M
      I2=1
30    IF(I1.EQ.I2) GO TO 40
      N=N+M
40    M=M*2
      IF(M.LE.L) GO TO 10
      K=L
      SEQWAL=BIWAL(N,X)
      RETURN
      END
```

```
      INTEGER FUNCTIØN BIWAL(K,X)
C
C     BIWAL GENERATES THE DESIRED FUNCTIØN AS PRØDUCTS
C     ØF RADEMACHER FUNCTIØNS.
C
      INTEGER RADFUN
      L=K
      M=1
      N=1
      I=-1
10    I=I+1
      IF(MØD(K,(2*M))) 30,30,20
20    N=N*RADFUN(I,X)
      K=K-M
30    M=M*2
      IF(M.LE.L) GØ TØ 10
      K=L
      BIWAL=N
      RETURN
      END
```

```
      INTEGER FUNCTIØN RADFUN(K,X)
C
C     RADFUN GENERATES THE RADEMACHER FUNCTIØNS USED
C     BY BIWAL.
C
      X=X-INT(X)
      N=INT(X*(2**(K+1)))
      IF(MØD(N,2)) 10,10,20
10    RADFUN=1
      RETURN
20    RADFUN=-1
      RETURN
      END
```

COMPUTER PROGRAM 2

Walsh Transform as Hadamard Matrix Product

```
C     ******************************************************************
C     *                                                                *
C     *   THIS PRØGRAM GENERATES A SET ØF WALSH FUNCTIØNS,             *
C     *   ARRANGES THEM INTØ A HADAMARD MATRIX, AND USES               *
C     *   IT TØ FIND THE /WALSH TRANSFØRM/ ØF AN INPUT VECTØR,         *
C     *   IT THEN MULTIPLIES THE TRANSFØRM BY A WEIGHTING              *
C     *   FUNCTIØN AND TAKES THE INVERSE TRANSFØRM, AGAIN USING        *
C     *   THE HADAMARD MATRIX.                                         *
C     *                                                                *
C     *   THE FIRST CARD ØF THE DATA CØNSISTS ØF FLAGS AND             *
C     *   THE ØRDER ØF THE DESIRED SYSTEM.  THE FLAGS                  *
C     *   DETERMINE THE FØRMAT ØF THE ØUTPUT AND ARE ENTERED AS        *
C     *   1 IF DESIRED AND 0 IF NØT DESIRED IN THE FIRST 6 CØL.        *
C     *   FLAG 1              PLØTS HADAMARD MATRIX                    *
C     *   FLAG 2              PLØTS INPUT (TIME DØMAIN)                *
C     *   FLAG 3              PLØTS INPUT (TRANSFØRM DØMAIN)           *
C     *   FLAG 4              PLØTS WEIGHTING FUNCTIØN                 *
C     *   FLAG 5              PLØTS ØUTPUT (TRANSFØRM DØMAIN)          *
C     *   FLAG 6              PLØTS ØUTPUT (TIME DØMAIN)               *
C     *   N  (MAX=128) ENTERED IN I3 FØRMAT CØLS. 10,11,12             *
C     *                                                                *
C     *   THE NEXT SET ØF CARDS CØNTAIN THE INPUT SAMPLES              *
C     *   N SAMPLES ENTERED 8 TØ A CARD IN F10.0 FØRMAT                *
C     *                                                                *
C     *   THE LAST SET ØF DATA CARDS CØNTAIN THE WEIGHTING             *
C     *   FUNCTIØN AND ARE ENTERED IN SAME FØRMAT AS INPUT,            *
C     *   N VALUES CØRRESPØNDING TØ THE N SPECTRAL CØMPØNENTS          *
C     *                                                                *
C     *   PRØGRAM WRITTEN IN IBM-360 SYSTEM FØRTRAN IV                 *
C     *                                                                *
C     ******************************************************************

      DIMENSIØN V(128),F(128)
      INTEGER*2 WAL(128,128),FLAG(8)
```

```
      WRITE (6,1000)
      READ(5,1001) (FLAG(I),I=1,6),N
      CALL WALSH(WAL,N)
      IF(FLAG(1).EQ.0) GØ TØ 10
      CALL HADPLT (WAL,V,N)
10    CALL RØR(F,N)
      IF(FLAG(2).EQ.0) GØ TØ 20
      FLAG(7)=0
      FLAG(8)=1
      CALL PLØTIT(F,N,FLAG(7),FLAG(8))
20    DØ 30 I=1,N
      V(I)=0.
      DØ 25 J=1,N
      A=WAL(I,J)
25    V(I)=(F(J)*A)+V(I)
30    V(I)=V(I)/FLØAT(N)
      IF(FLAG(3).EQ.0) GØ TØ 40
      FLAG(7) =0
      FLAG(8)=0
      CALL PLØTIT(V,N,FLAG(7),FLAG(8))
40    CALL RØR(F,N)
      IF(FLAG(4).EQ.0) GØ TØ 50
      FLAG(7)=-1
      CALL PLØTIT(F,N,FLAG(7),FLAG(8))
50    DØ 60 I=1,N
60    V(I)=V(I)*F(I)
      IF (FLAG(5).EQ.0) GØ TØ 65
      FLAG(7)=1
      FLAG(8)=0
      CALL PLØTIT(V,N,FLAG(7),FLAG(8))
65    DØ 70 I=1,N
      F(I)=0.
      DØ 70 J=1,N
      A=WAL(I,J)
70    F(I)=(V(J)*A)+F(I)
```

```
      IF(FLAG(6).EQ.0) GO TO 80
      FLAG(7)=1
      FLAG(8)=1
      CALL PLOTIT(F,N,FLAG(7),FLAG(8))
80    STOP
1000  FORMAT(1H1)
1001  FORMAT(6I1,3X,I3)
      END
```

```
      SUBROUTINE WALSH(W,N)
C     *
C     *   WALSH GENERATES A SET OF WALSH FUNCTIONS USING
C     *   THE METHOD DESCRIBED BY PETERSON.
C     *
      INTEGER*2 W(128,128),B(7)
      LN=INT((ALOG(FLOAT(N))/ALOG(2.))+.1)
      DO 50 I=1,N
      K=I-1
      J=0
5     J1=J
      J=J+1
      IF(MOD(K,2))20,20,10
10    K=K-1
      B(J)=-1
      GOTO30
20    B(J)=1
30    K=K/2
      IF(J.LE.1)GOTO5
      B(J1)=B(J)*B(J1)
      IF(LN-J)40,40,5
40    W(I,1)=1
      DO50 J=1,LN
      JZ=LN-J+1
      K=2**(J-1)
      DO50 JX=1,K
      JY=JX+K
50    W(I,JY)=W(I,JX)*B(JZ)
      RETURN
      END
```

```
      SUBROUTINE HADPLT(WAL,V,N)
C     *
C     *   HADPLT DISPLAYS THE HADAMARD MATRIX
C     *   AS A SET OF +'S AND -'S.
C     *
      INTEGER*2 WAL(128,128)
      DIMENSION V(128)
      REAL MINUS
      DATA PLUS,MINUS,BLANK/1H+,1H-,1H /
      WRITE (6,100) N
      DO 50 I=1,N
      DO 40 J=1,N
      IF(WAL(I,J)) 10,20,30
10    V(J)=MINUS
      GO TO 40
20    V(J)=BLANK
      GO TO 40
30    V(J)=PLUS
40    CONTINUE
50    WRITE(6,101) (V(J),J=1,N)
      WRITE(6,102)
      RETURN
100   FORMAT(' GENERATED HADAMARD MATRIX OF ORDER ',I3,//)
101   FORMAT(' ',128A1)
102   FORMAT(1H1)
      END
```

```
      SUBROUTINE PLOTIT (V,N,N7,N8)
C     *
C     *   PLOTIT DISPLAYS DESIRED OUTPUTS IN BAR GRAPH FORM
C     *
      INTEGER*2 N7,N8
      DIMENSION V(128),L(128),O(50)
      DATA STAR,PLOT,BLANK/1H*,1H+,1H /
      IF (N7) 10,11,12
10    WRITE(6,104)
      GO TO 20
11    WRITE(6,105)
      GO TO 13
12    WRITE(6,106)
13    IF(N8) 15,15,14
14    WRITE(6,107)
      GO TO 20
15    WRITE(6,108)
20    WRITE(6,109)
      X=0.
      DO 25 I=1,N
25    X=AMAX1((ABS(V(I))),X)
      IF(X.EQ.0.) GO TO 90
      DO 30 I=1,N
30    L(I)=INT((50.*(V(I)/X))+.5)
      DO 85 I=1,N
      I1=I-1
      M=1
      IF(L(I)) 35,70,60
35    M=50+L(I)
      IF(M.EQ.0) GO TO 45
      DO 40 J=1,M
40    O(J)=BLANK
45    M=M+1
      O(M)=STAR
      IF(M.EQ.50) GO TO 55
```

```
      SUBRØUTINE RDR(A,N)
C     *
C     *   RDR IS USED TØ READ INPUT DATA FRØM CARDS
C     *
      DIMENSIØN A(N)
      NA=N/8
      DØ 10 I=1,NA
      JA=((I-1)*8)+1
      JB=I*8
10    READ(5,100) (A(J),J=JA,JB)
      RETURN
100   FØRMAT(8F10.0)
      END
```

```
      M=M+1
      DO 50 J=M,50
50    O(J)=PLOT
55    WRITE(6,100) I1,V(I),(O(J),J=1,50),STAR
      GO TO 85
60    M=L(I)
      DO 65 J=1,M
65    O(J)=PLOT
      O(M)=STAR
      IF(M.EQ.50) GO TO 80
      M=M+1
70    DO 75 J=M,50
75    O(J)=BLANK
80    WRITE(6,101) I1,V(I),STAR,(O(J),J=1,50)
85    CONTINUE
      WRITE(6,102)
      RETURN
90    WRITE(6,103)
      STOP
100   FORMAT(' ',I3,1X,E12.6,1X,51A1)
101   FORMAT(' ',I3,1X,E12.6,51X,51A1)
102   FORMAT(1H1)
103   FORMAT(///,30X,' N U L L   A R R A Y ',////,' EXECUTION TERMINATED
     1',/,1H1)
104   FORMAT(40X,' WEIGHTING FUNCTION')
105   FORMAT(41X,' INPUT')
106   FORMAT(41X,' OUTPUT')
107   FORMAT(35X,' IN THE TIME DOMAIN')
108   FORMAT(35X,' IN THE TRANSFORM DOMAIN')
109   FORMAT(7X,'MAGNITUDE',45X,'NORMALIZED PLOT')
      END
```

COMPUTER PROGRAM 3

Fast Walsh Transform

```
C     ****************************************************************
C     *                                                              *
C     *   THIS PRØGRAM FINDS THE WALSH TRANSFØRM ØF AN INPUT         *
C     *   VECTØR USING A JFAST TRANSFØRMJ ALGØRITHIM.  IT THEN       *
C     *   MULTIPLIES THE TRANSFØRM BY A WEIGHTING FUNCTIØN AND       *
C     *   TAKES THE INVERSE TRANSFØRM USING THE SAME ALGØRITHIM.     *
C     *                                                              *
C     *   THE FIRST CARD ØF THE DATA CØNSISTS ØF FLAGS AND           *
C     *   THE ØRDER ØF THE DESIRED SYSTEM.  THE FLAGS                *
C     *   DETERMINE THE FØRMAT ØF THE ØUTPUT AND ARE ENTERED AS      *
C     *   1 IF DESIRED AND 0 IF NØT DESIRED IN THE FIRST 5 CØL.      *
C     *   FLAG 1              PLØTS INPUT (TIME DØMAIN)              *
C     *   FLAG 2              PLØTS INPUT (TRANSFØRM DØMAIN)         *
C     *   FLAG 3              PLØTS WEIGHTING FUNCTIØN               *
C     *   FLAG 4              PLØTS ØUTPUT (TRANSFØRM DØMAIN)        *
C     *   FLAG 5              PLØTS ØUTPUT (TIME DØMAIN)             *
C     *   N  (MAX=128) ENTRED IN I3 FØRMAT CØLS. 10,11,12            *
C     *                                                              *
C     *   THE NEXT SET ØF CARDS CØNTAIN THE INPUT SAMPLES            *
C     *   N SAMPLES ENTERED 8 TØ A CARD IN F10.0 FØRMAT              *
C     *                                                              *
C     *   THE LAST SET ØF DATA CARDS CØNTAIN THE WEIGHTING           *
C     *   FUNCTIØN AND ARE ENTERED IN SAME FØRMAT AS INPUT,          *
C     *   N VALUES CØRRESPØNDING TØ THE N SPECTRAL CØMPØNENTS        *
C     *                                                              *
C     *   NØTES:                                                     *
C     *     (1) N(MAX) CAN BE INCREASED TØ ANY SIZE WITHIN           *
C     *     CØMPUTER LIMITATIØNS BY REDIMENSIØNING V(NMAX),          *
C     *     F(NMAX),T(NMAX), AND B(LØG BASE 2 NMAX).                 *
C     *     (2) PRØGRAM LANGUAGE IS IBM-360 SYSTEM FØRTRAN IV        *
C     *                                                              *
C     ****************************************************************
C
      DIMENSIØN V(128),F(128),T(128)
      INTEGER*2 FLAG(8),B(8)
```

```
      WRITE (6,1000)
      READ(5,1001) (FLAG(I),I=2,6),N
      LN=INT(((ALØG(FLØAT(N)))/(ALØG(2.)))+.1)
      CALL RCR(F,N)
      IF(FLAG(2).EQ.0) GØ TØ 20
      FLAG(7)=0
      FLAG(8)=1
      CALL PLØTIT(F,N,FLAG(7),FLAG(8),T)
20    FLAG(1)=1
      CALL FASWAL(F,V,N,LN,FLAG(1),T,B)
      IF(FLAG(3).EQ.0) GØ TØ 40
      FLAG(7) =0
      FLAG(8)=0
      CALL PLØTIT(V,N,FLAG(7),FLAG(8),T)
40    CALL RCR(F,N)
      IF(FLAG(4).EQ.0) GØ TØ 50
      FLAG(7)=-1
      CALL PLØTIT(F,N,FLAG(7),FLAG(8),T)
50    DØ 60 I=1,N
60    V(I)=V(I)*F(I)
      IF (FLAG(5).EQ.0) GØ TØ 65
      FLAG(7)=1
      FLAG(8)=0
      CALL PLØTIT(V,N,FLAG(7),FLAG(8),T)
65    FLAG(1)=0
      CALL FASWAL(V,F,N,LN,FLAG(1),T,B)
      IF(FLAG(6).EQ.0) GØ TØ 80
      FLAG(7)=1
      FLAG(8)=1
      CALL PLØTIT(F,N,FLAG(7),FLAG(8),T)
80    STØP
1000  FØRMAT(1H1)
1001  FØRMAT(5I1,4X,I3)
      END
```

```
      SUBRØUTINE FASWAL(F,X,N,LN,IX,T,B)
C     *
C     *   FASWAL FINDS THE WALSH TRANSFØRM ØR INVERSE TRANSFØRM
C     *   ØF AN INPUT VECTØR USING A VFAST TRANSFØRMV ALGØRITHIM
C     *
C     *   DESCRIPTIØN ØF VARIABLES:
C     *        F - INPUT VECTØR ØF LENGTH N
C     *        X - ØUTPUT VECTØR ØF LENGTH N
C     *        N - ØRDER ØF SYSTEM
C     *        LN - LØG BASE 2 ØF N
C     *        T - INTERNAL WØRKING VECTØR LENGTH N
C     *        B - INTERNAL WØRKING VECTØR ØF LENGTH LN
C     *        IX - FØR TRANSFØRM, IX=1
C     *             FØR INVERSE XFØRM, IX=0
C     *
C     *   FASWAL REQUIRES INTEGER FUNCTIØN SUBRØUTINE CØNVRT
C     *
C     *****************************************************************
      DIMENSIØN F(N),X(N),T(N)
      INTEGER*2 IX,B(LN)
      INTEGER CØNVRT
      N1=N/2
      DØ 10 I=1,N
10    T(I)=F(I)
      DØ 30 J=1,LN
      DØ 20 K=1,N1
      K1=2*K
      X(K)=T(K1-1)+T(K1)
      K2=K+N1
20    X(K2)=T(K1-1)-T(K1)
      DØ 30 I=1,N
30    T(I)=X(I)
C
C     T(I) NØW CØNTAINS RESULT IN REVERSE REFLECTED BINARY ØRDER
C
```

```
      NIX=N**IX
      DO 40 J=1,N
      JX=J-1
40    X(J)=T(CONVRT(JX,N,LN,B)+1)/NIX
      RETURN
      END
```

```
      INTEGER FUNCTION CONVRT(J,N,LN,B)
C     *
C     *   CONVRT CHANGES THE REVERSE REFLECTED BINARY INDEX
C     *   INTO SEQUENCY ORDERED INDEX
C     *
C     ***********************************************************
      INTEGER*2 B(LN)
      I=0
      K=J
5     I1=I
      I=I+1
      IF(MOD(K,2)) 20,20,10
10    K=K-1
      B(I)=1
      GO TO 30
20    B(I)=0
30    K=K/2
      IF(I.LE.1)GOTO5
      IB=B(I1)+B(I)
      B(I1)=MOD(IB,2)
      IF(LN-I) 40,40,5
40    CONVRT=0
      DO 50 I=1,LN
50    CONVRT=CONVRT+(B(I)*(2**(LN-I)))
      RETURN
      END
```

```
      SUBRØUTINE RDR(A,N)
C     *
C     *   RDR IS USED TØ READ INPUT DATA FRØM CARDS
C     *
      DIMENSIØN A(N)
      NA=N/8
      DØ 10 I=1,NA
      JA=((I-1)*8)+1
      JB=I*8
10    READ(5,100) (A(J),J=JA,JB)
      RETURN
100   FØRMAT(8F10.0)
      END
```

```
      SUBRØUTINE PLØTIT (V,N,N7,N8,L)
C     *
C     *   PLØTIT DISPLAYS ALL ØUTPUTS DESIRED IN A BAR GRAPH FØRM
C     *
      INTEGER*2 N7,N8
      DIMENSION V(N),L(N),B(50)
      DATA STAR,PLØT,BLANK/1H*,1H+,1H /
      IF (N7) 10,11,12
10    WRITE(6,104)
      GØ TØ 20
11    WRITE(6,105)
      GØ TØ 13
12    WRITE(6,106)
13    IF(N8) 15,15,14
14    WRITE(6,107)
      GØ TØ 20
15    WRITE(6,108)
20    WRITE(6,109)
      X=0.
      DØ 25 I=1,N
25    X=AMAX1((ABS(V(I))),X)
      IF(X.EQ.0.) GØ TØ 90
      DØ 30 I=1,N
30    L(I)=INT((50.*(V(I)/X))+.5)
      DØ 85 I=1,N
      I1=I-1
      M=1
      IF(L(I)) 35,70,60
35    M=50+L(I)
      IF(M.EQ.0) GØ TØ 45
      DØ 40 J=1,M
40    B(J)=BLANK
45    M=M+1
      B(M)=STAR
      IF(M.EQ.50) GØ TØ 55
```

```
        M=M+1
        DØ 50 J=M,50
50      B(J)=PLØT
55      WRITE(6,100) I1,V(I),(B(J),J=1,50),STAR
        GØ TØ 85
60      M=L(I)
        DØ 65 J=1,M
65      B(J)=PLØT
        B(M)=STAR
        IF(I.EQ.50) GØ TØ 80
        M=M+1
70      DØ 75 J=M,50
75      B(J)=BLANK
80      WRITE(6,101) I1,V(I),STAR,(B(J),J=1,50)
85      CØNTINUE
        WRITE(6,102)
        RETURN
90      WRITE(6,103)
        STØP
100     FØRMAT(' ',I3,1X,E12.6,1X,51A1)
101     FØRMAT(' ',I3,1X,E12.6,51X,51A1)
102     FØRMAT(1H1)
103     FØRMAT(///,30X,' N U L L   A R R A Y ',////,' EXECUTIØN TERMINATED
       1',/,1H1)
104     FØRMAT(40X,' WEIGHTING FUNCTIØN')
105     FØRMAT(41X,' INPUT')
106     FØRMAT(41X,' ØUTPUT')
107     FØRMAT(35X,' IN THE TIME DØMAIN')
108     FØRMAT(35X,' IN THE TRANSFØRM DØMAIN')
109     FØRMAT(7X,'MAGNITUDE',45X,'NØRMALIZED PLØT')
        END
```

BIBLIOGRAPHY

1. Andrews, H. C., "Digital Image Processing," Symposium on Walsh Functions, Naval Research Laboratory, Wash., D. C., 1970.

2. Andrews, H. C. and Caspari, K. L., "A Generalized Technique for Spectral Analysis," IEEE Trans. on Computers, Vol. C-19, No. 1, pp. 16-25, January 1970.

3. Andrews, H. C. and Pratt, W. K., "Application of Fourier-Hadamard Transformation to Bandwidth Compression," Symposium on Picture Bandwidth Compression, MIT, April, 1969.

4. Ballard, A. H., "Orthogonal Multiplexing," Space Aeronautics, Part 2, pp. 50-60, November 1962.

5. Byrnes, J. S. and Swick, D. A., "Instant Walsh Functions," SIAM Review, January 1970.

6. Carl, J. W., "An Application of Walsh Functions to Image Classification," Symposium on Walsh Functions, Naval Research Laboratory, Wash., D. C.,1970.

7. Corrington, M. S., and Adams, R. N., "Advanced Analytical and Signal Processing Techniques; Application of Walsh Functions to Nonlinear Analysis," Technical Report, ASTIA #AD-277942, 1962.

8. Gold, B. and Rader, C. M., Digital Processing of Signals, McGraw-Hill, New York, 1969.

9. Harmuth, H. F., Transmission of Information by Orthogonal Functions, Springler-Verlag, Berlin/New York, 1969.

10. Harr, A., "Zur Theorie der orthogonalen Funktionensysteme," Erste Mitteilung. Math. Ann., Vol. 69, pp 331-371, 1909.

11. Hubner, H., et al, "Algorithm 388 - Rademacher Function," Comm. ACM 13, p. 510, August 1970.

12. Hubner, H., et al, "Algorithm 389 - Binary Ordered Walsh Functions," Comm. ACM 13, p. 511, August 1970.

13. Hubner, H., et al, "Algorithm 390 - Sequency Ordered Walsh Functions," Comm. ACM 13, p. 512, August 1970.

14. Huebner, H., "Analog and Digital Multiplexing by Means of Walsh Functions," Symposium on Walsh Functions, Naval Research Laboratory, Wash., D.C., 1970.

15. Huebner, H., "On the Transmission of Walsh Multiplexed Signals," Symposium on Walsh Functions, Naval Research Laboratory, Wash., D.C., 1970.

16. Kennett, B. L. N., "A Note on the Finite Walsh Transform," IEEE Trans. Info. Theory, IT-6, pp 489-491, July 1970.

17. Lueke, H. D., "Binary Orthogonal Functions in Communications," Internationale Elecktronische Rundschau, Vol. 29, No. 1, pp 1-3, January 1970.

18. Peterson, H. L., "Generation of Walsh Functions," Symposium on Walsh Functions, Naval Research Laboratory, Wash., D. C., 1970.

19. Pratt, W. K., and Andrews, H. C., "Hadamard Transform Image Coding," Proceedings of the IEEE, Vol. 57, No. 1, pp 58-68, January 1969.

20. Rademacher, H., "Einige Sätze über Reihen von allegemeinen Orthogonalfunktionen," Math. Ann., Vol. 87, pp 112-138, 1922.

21. Shanks, J. L., "Computation of the Fast Walsh-Fourier Transform," IEEE Trans. on Computers, pp 457-459, May 1969.

22. Swick, D. A., "Walsh Function Generation," IEEE Trans. on Info. Theory, Vol. IT-15, No. 1, p. 167, January 1969.

23. Vandivere, E. F., and Carrick, R. L., "Fast Walsh Transform," Telcom Inc. Report TAM 2.1.2, 13 November 1967.

24. Walsh, J. L., "A Closed Set of Normal Orthogonal Functions," Amer. Jour. Math., Vol. 45, pp 5-24, 1923.

25. Whelchel, J. E. and Guinn, D. F., "Fast Fourier-Hadamard Transform and its Use in Signal Representation and Classification," EASCON Record, pp 561-573, 1968.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Dr. S. R. Parker, Code 52 Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	2
4.	William J. Dejka, Code 4300 Advanced Modular Concepts Division Head Naval Electronics Laboratory Center San Diego, California 92152	1
5.	Lt. Norman Clare Meck, USN P. O. Box 67 Palmer Lake, Colorado 80133	1

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Naval Postgraduate School Monterey, California 93940	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

Application of Discrete Walsh Functions to Digital Filter Techniques

4. DESCRIPTIVE NOTES (Type of report and, inclusive dates)

Electrical Engineer's Thesis; June 1972

5. AUTHOR(S) (First name, middle initial, last name)

Norman C. Meck

6. REPORT DATE June 1972	7a. TOTAL NO. OF PAGES 80	7b. NO. OF REFS 25
8a. CONTRACT OR GRANT NO. b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	

10. DISTRIBUTION STATEMENT

Approved for public release, distribution unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Naval Postgraduate School Monterey, California 93940

13. ABSTRACT

Methods of generating and properties of Walsh functions are discussed with emphasis on discrete Walsh functions. Hardware and software versions of two methods of Walsh function generation are shown. The Walsh transform is explored and hardware and software realizations of a fast Walsh transform algorithm, particularly applicable to hardware, is introduced. Computer programs containing the various algorithms are included. A discussion of the feasibility of the use of Walsh functions to produce a digital filter with a desired frequency response is presented.

Security Classification

KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Walsh Function						
Walsh Transform						
Harr Function						
Rademacher Function						
Hadamard Transform						
Walsh Fourier Transform						
Digital Filter						

FORM 1 NOV 65 1473 (BACK)

101-807-6821

Security Classification

A-31409

Thesis 135464
M42 Meck
c.1 Application of discrete
Walsh functions to digi-
tal filter techniques.

10 AUG 73 21118
30 JUN 76 23472
[illegible]
16 JUL 87 14455
27 July 86 30631

Thesis 135464
M42 Meck
c.1 Application of discrete
Walsh functions to digi-
tal filter techniques.

thesM42
Application of discrete Walsh functions
3 2768 000 98315 9
DUDLEY KNOX LIBRARY